RFID COMMANDS MANUAL

Code: **DOMC-0002e**

Edit by:

CUSTOM ENGINEERING S.p.A.
Str. Berettine 2 - 43010 Fontevivo (PARMA) - Italy
http: www.custom.biz

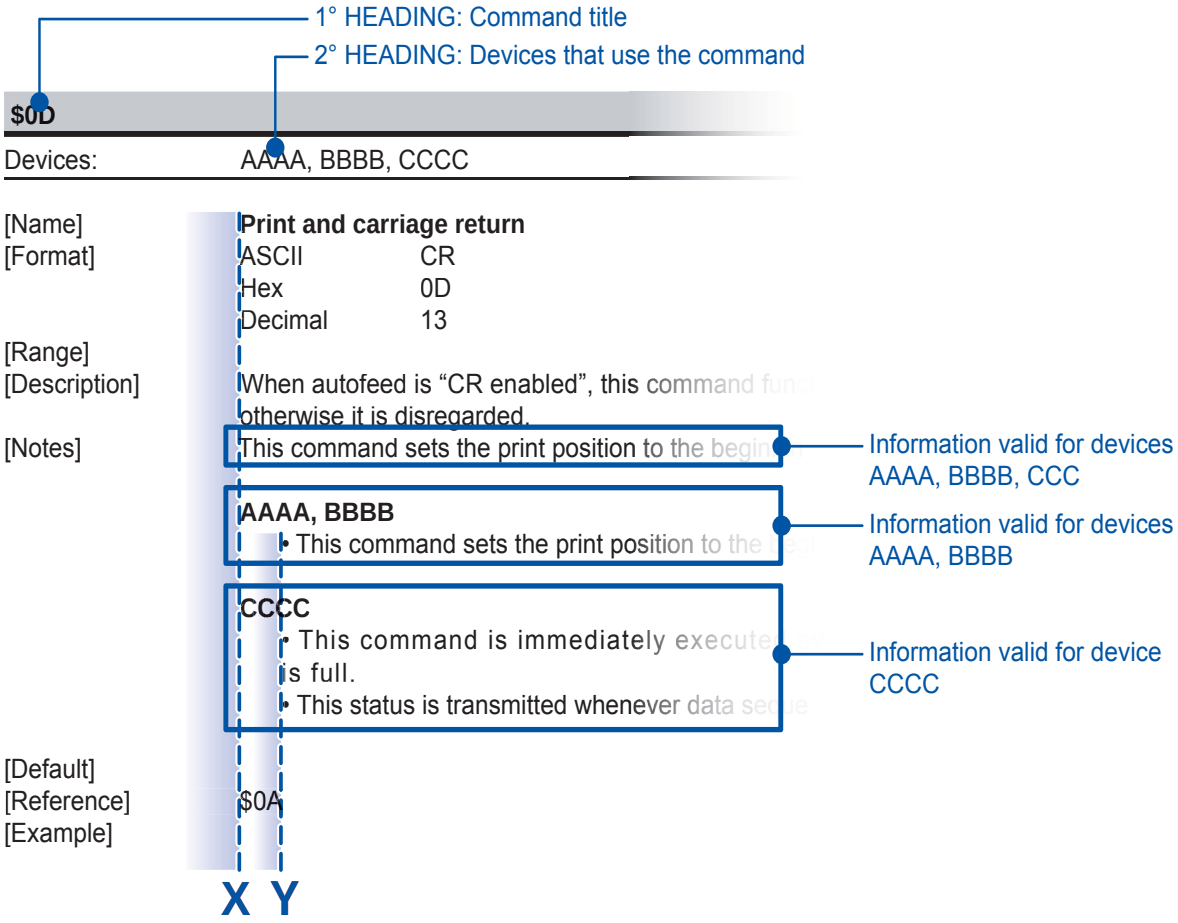
All rights reserved

1 INTRODUCTION

1.1 CUSTOM command description

Each Custom command reported in this manual is described as shown in the following picture. In the first heading line (grey colour) is reported the hexadecimal command value. In the second heading line are listed the devices on which it is possible to use the command (for example device AAAA).
The next fields give all the information useful to use the command.

[Name]	Command title
[Format]	ASCII, hexadecimal and decimal command value.
[Range]	Limits of the values the command and its variables can take
[Description]	Description of command function
[Notes]	Additional information about command use and settings .
[Default]	Default value of the command and its variables.
[Reference]	Pertaining commands related to described command.
[Example]	

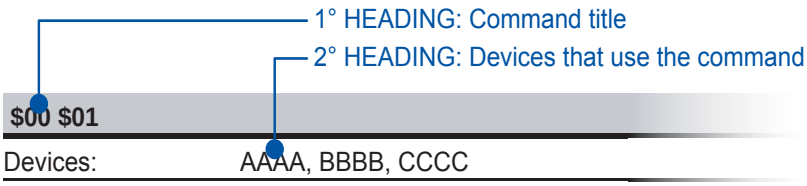

1° HEADING: Command title
2° HEADING: Devices that use the command
$0D
Devices: AAAA, BBBB, CCCC
[Name] Print and carriage return
[Format] ASCII CR
Hex 0D
Decimal 13
[Range]
[Description] When autofeed is “CR enabled”, this command
otherwise it is disregarded.
[Notes] This command sets the print position to the
Information valid for devices AAAA, BBBB, CCC
AAAA, BBBB
• This command sets the print position to the
Information valid for devices AAAA, BBBB
CCCC
• This command is immediately
is full.
• This status is transmitted whenever data
Information valid for device CCCC
[Default]
[Reference] $0A
[Example]
X Y

The information reported in the picture are aligned with line X or line Y:	
LINE X	Description valid for all the devices listed in the second heading line.
LINE Y	Description valid for a specific device (written in bold).

LEGEND	
$	indicates the representation of the command hexadecimal value (for example $40 means HEX 40).
{ }	indicates an ASCII character not performable.
n, m, t, x, y	are optional parameters that can have different values.

1.2 ASK command description

All functions of the ASK reader will be presented as follows. In the first heading line (grey colour) is reported the hexadecimal command value. In the second heading line are listed the devices on which it is possible to use the command (for example device AAAA).
The next fields give all the information useful to use the command.



Name : Command name
Description : Actions carried out by the command

CLASS	INS	DATA IN		
[Class]	*[Ins]*	*[P1] (a)*	*[P2]*	...

[Px] : list of input data
a : Length in bytes of data (x = variable)

CLASS	INS	DATA OUT		
[Class]	*[Ins]*	*[D1] (a)*	*[D2]*	...

[Dx] : list of output data
a : Length in bytes of data (x = variable)

LEGEND	
$	indicates the representation of the command hexadecimal value (for example $40 means HEX 40).

2 HF CUSTOM COMMANDS

The following table lists all the commands for function management in ESC/POS Emulation of the printer. The commands can be transmitted to the printer at any moment, but they will only be carried out when the commands ahead of them have been executed. The commands are carried out when the circular buffer is free to do so.

COMMAND DESCRIPTION TABLE

Com. HEX	Com. ASCII	Description
PASS THROUGH COMMANDS		
$1B $3D	ESC =	Enable/Disable the communication with RF module
ISO14443-A & MIFARE COMMANDS		
$1F $30	US 0	Reset the PCD
$1F $34	US 4	Write the 4 bytes of the TAG sector
$1F $43	US C	PCD setting for ISO14443 communication
$1F $45	US E	Read/write EEPROM memory of PCD
$1F $49	US I	Read identification code
$1F $4C	US L	Save key in EEPROM
$1F $52	US R	Switch off the RF carrier for n milliseconds
$1F $53	US S	RFID status
$1F $58	US X	Data Exchange with PICC
$1F $61	US a	Mifare Anticollision
$1F $62	US b	Mifare Authentication, keys EEPROM
$1F $63	US c	ISO14443-A Anticollision
$1F $64	US d	ISO14443-A Select
$1F $68	US h	Mifare Halt Card
$1F $6B	US k	Mifare Authentication with keys
$1F $71	US q	Mifare request
$1F $72	US r	Mifare read
$1F $73	US s	Mifare select
$1F $75	US u	Mifare ultralight multiple Write
$1F $76	US v	Mifare ultralight multiple Read
$1F $77	US w	Mifare write
ISO15693 & I-CODE COMMANDS		
$1E $43	RS C	Set the PCD for ISO15693
$1E $44	RS D	Initialize SLI ISO15693 mode
$1E $45	RS E	Initialize ICODE1 mode
$1E $49	RS I	Inventory ISO15693
$1E $52	RS R	Switch off the RF carrier for n milliseconds
$1E $57	RS W	Write
$1E $61	RS a	Anticollision
$1E $72	RS r	Read
$1E $73	RS s	Return the UID
$1E $FF	RS { }	Read/write a value in the PCD register

2.1 Commands detailed descriptions

Given below are more detailed descriptions of each command.

$1B $3D

Devices: *all printer models with RFID reader/writer*

[Name] **Pass Through**

[Format]

ASCII	ESC	=	n
Hex	1B	3D	n
Decimal	27	61	n

[Description]
n = 1 Disable the communication with RF module
n = 5 Enable the communication with RF module

[Notes] For using commands RF, it's necessary that the communication with RF module is enabled. For using normal commands ESC/POSTM, it's necessary disable the communication with RF.

[Default] n = 1

[Reference]

[Example]

$1E $43

Devices: *all printer models with RFID reader/writer*

[Name] **Set the PCD for ISO 15693**

[Format]

ASCII	RS	C
Hex	1E	43
Decimal	30	67

[Description] Set the PCD for the ISO15693 protocol.
Returns:
Status.

[Notes]

[Default]

[Reference]

[Example]

$1E $44

Devices: *all printer models with RFID reader/writer*

[Name] **Initialize SLI ISO 15693 mode**

[Format]

ASCII	RS	D	p
Hex	1E	44	p
Decimal	30	68	p

[Description] Initialize SLI ISO 15693 communication mode:

p = $53 Standard mode
p = $46 Fast mode

Returns:
Status.

[Notes]
[Default]
[Reference]
[Example]

$1E $45

Devices: *all printer models with RFID reader/writer*

[Name] **Initialize I-CODE1 mode**

[Format]

ASCII	RS	E	p
Hex	1E	45	p
Decimal	30	69	p

[Description] Initialize I-CODE1 communication mode:

p = $53 Standard mode
p = $46 Fast mode

Returns:
Status.

[Notes]
[Default]
[Reference]
[Example]

$1E $49

Devices: *all printer models with RFID reader/writer*

[Name] **Inventory ISO 15693**

[Format]

ASCII	RS	I	p	a	n	u0…ul
Hex	1E	49	p	a	n	u0…ul
Decimal	30	73	p	a	n	u0…ul

[Description] Send a request to all the TAGs present in RF.
Answer in accordance with the modality defined in the byte p (flag) the TAGs having AFI (Application Family Identifier) specified from a and having the first n bits of the UID coinciding with the bits in u=…ul.
p = FLAGS
a = AFI
n = number of bits known
u0…ul = Known byte of the UID

Returns:
status, Num, Resp.

Where
Num = number of bytes of the answer
Resp = answer codified with ISO/IEC 15693

[Notes]
[Default]
[Reference]
[Example]

$1E $52

Devices: *all printer models with RFID reader/writer*

[Name] **Switch off the RF carrier for n milliseconds**

[Format]

ASCII	RS	R	n
Hex	1E	52	n
Decimal	30	82	n

[Description] Carrier signal for the communication in RF is not transmitted for n milliseconds. In this mode the TAG on the antenna is reset.
If n = 0 the RF transmission is switched off permanently.
1 ≤ n ≤ 255 Switch off period for the antenna

[Notes]
[Default]
[Reference]
[Example]

$1E $57

Devices: *all printer models with RFID reader/writer*

[Name] **Write**

[Format]

ASCII	RS	W	p	u0…u7 n	a	d0…d3
Hex	1E	57	p	u0…u7 n	a	d0…d3
Decimal	30	87	p	u0…u7 n	a	d0…d3

[Description] Write the 4 bytes data
p = flags
u0…u7 single UID
n = number of the first block to write
a = number of blocks to write (For ICODE SLI a = 1)
d0…d3 = data to write.

[Notes]
[Default]
[Reference]
[Example]

$1E $61

Devices: *all printer models with RFID reader/writer*

[Name] **Anticollision**

[Format]

ASCII	RS	a	p	n
Hex	1E	61	p	n
Decimal	30	97	p	n

[Description] Returns the number of TAGs present in RF
p = flags
n = AFI

Returns:
Number of TAGs present in RF

[Notes]
[Default]
[Reference]
[Example]

$1E $72

Devices: *all printer models with RFID reader/writer*

[Name] **Read**

[Format]

ASCII	RS	r	p	u0…u7 n	a
Hex	1E	72	p	u0…u7 n	a
Decimal	30	114	p	u0…u7 n	a

[Description] Read a number of blocks equal to a 4 bytes block starting from block number n
p = flags
u0…u7 single UID
n = number of the first block to read
a = = number of blocks to read

[Notes]
[Default]
[Reference]
[Example]

$1E $73

Devices: *all printer models with RFID reader/writer*

[Name] **Returns UID**

[Format]

ASCII	RS	s	n
Hex	1E	73	n
Decimal	30	115	n

[Description] Returns the UID of the TAG number n
n = number of the TAG to read for UID

Return:

[Notes]
[Default]
[Reference]
[Example]

$1E $FF

Devices: *all printer models with RFID reader/writer*

[Name] **Read/write a value in the PCD register**

[Format]

ASCII	RS	{ }	n	a	d *
Hex	1E	FF	n	a	d *
Decimal	30	255	n	a	d *

[Description] Read or write in a PCD register having address a the value d following tha table below:

n	a	d *	Operation
$52	Addr		Read
$57	Addr	data	Write

[Notes] * The data to write in the register must be specified only during the writing operation.

[Default]

[Reference]

[Example]

$1F $30

Devices: *all printer models with RFID reader/writer*

[Name] **Reset the PCD**

[Format]

ASCII	US	0
Hex	1F	30
Decimal	31	48

[Description] Reset the RF device used for the communication.

[Notes]

[Default]

[Reference]

[Example]

$1F $34

Devices: *all printer models with RFID reader/writer*

[Name] **Write the 4 bytes of the TAG sector (MfUL)**

[Format]

ASCII	US	4	n	b0 … b3
Hex	1F	34	n	b0 … b3
Decimal	31	52	n	b0 … b3

[Description] Writes the 4 bytes b0, b1, b2, b3 of the page number n of TAG.

0 ≤ n ≤ 15
0 ≤ b ≤ 255

return
Status.

[Notes]
[Default]
[Reference]
[Example]

$1F $43

Devices: *all printer models with RFID reader/writer*

[Name] **PCD setting for ISO14443 communication**

[Format]

ASCII	US	C
Hex	1F	43
Decimal	31	67

[Description] Set the registers of the RF module.
[Notes]
[Default]
[Reference]
[Example]

$1F $45

Devices: *all printer models with RFID reader/writer*

[Name] **Read/write EEPROM memory of PCD**

[Format]

ASCII	US	E	n	a1	a0	L	d1…dL*
Hex	1F	45	n	a1	a0	L	d1…dL*
Decimal	31	69	n	a1	a0	L	d1…dL*

[Description] Read/write EEPROM memory of PCD a number of bytes L from address 1 to 0
a1 = Starting Address byte High
a0 = Starting Address byte Low

n	a1 a0	L	Data	Operation
$52	AddrH AddrL	Length		Read
$57	AddrH AddrL	Length	d1 …dL	Write

1≤ L ≤ 16 = number of bytes to read and write.

[Notes] * Data to write must be specified only during writing operation
[Default]
[References]
[Example]

$1F $49

Devices: *all printer models with RFID reader/writer*

[Name] **Read identification code (Mf1K, Mf4K, MfUL)**

[Format]

ASCII	US	I
Hex	1F	49
Decimal	31	73

[Description] Command that inquire to TAG the identification code (UID)
Return:

Status	Level	UID	Size
Status	$04	4 bytes	SAK *
Status	$07	7 bytes	SAK *

[Notes] * SAK Select Acknowledge (1 byte)
[Default]
[Reference]
[Example]

$1F $4C

Devices: *all printer models with RFID reader/writer*

[Name] **Save key in EEPROM of PCD**

[Format]

ASCII	US	L	t	s	k0…k5
Hex	1F	4C	t	s	k0…k5
Decimal	31	76	t	s	k0…k5

[Description] Save the key like t of the sector s.
This command is used for Mifare TAG with key (example Mf1K, Mf4K).

t = $41 key like A
t = $42 key like B

0 ≤ s ≤ 15 = number of keys to write

k0…K5 = 6 bytes including the key not crypted

Return:
status

[Notes]
[Default]
[Reference]
[Example]

$1F $52

Devices: *all printer models with RFID reader/writer*

[Name] **Switch off the RF carrier for n milliseconds (Mf1K, Mf4K, MfUL)**

[Format]

ASCII	US	R	n
Hex	1F	52	n
Decimal	31	82	n

[Description] The RF carrier isn't transferred for a n milliseconds. In this way the TAG on the antenna will be reset.
If n = 0 the RF transmission is switched on permanently
1 ≤ n ≤ 255 = antenna switched on time.

Return:
status

[Notes]
[Default]
[Reference]
[Example]

$1F $53

Devices: *all printer models with RFID reader/writer*

[Name] **Returns PCD status (Mf1K, Mf4K, MfUL)**

[Format]

ASCII	US	S
Hex	1F	53
Decimal	31	83

[Description] Return 4 bytes of the PCD register

b0 = RegErrorFlag
b1 = RegPrimaryStatus
b2 = RegSecondaryStatus
b3 = RegInterruptRq

[Notes]
[Default]
[Reference]
[Example]

$1F $58

Devices: *all printer models with RFID reader/writer*

[Name] **Data Exchange with PICC CC (Mf1K, Mf4K, MfUL)**

[Format]

ASCII	US	X	nH	nL	CRC	t0…t3	d0…dn
Hex	1F	58	nH	nL	CRC	t0…t3	d0…dn
Decimal	31	88	nH	nL	CRC	t0…t3	d0…dn

[Description] Transfer into the memory of the TAG the DATA d0…dn of length n.

CRC = 1 After the sequence of the datas, 2 bytes of CRC are on
CRC = 0 CRC not disable

t0, t1 show the high word of the dubleword value of received timeout
t2, t3 shows the low word of the dubleword value of received timeout

d0…dn indicate the data to transmit.

Return :
Status, l , d0..dl

l = bytes number of the data to receive
d…dl = data received

[Notes]
[Default]
[Reference]
[Example]

$1F $61

Devices: *all printer models with RFID reader/writer*

[Name] **Mifare Anticollision (Mf1K, Mf4K, MfUL)**

[Format]

ASCII	US	S
Hex	1F	61
Decimal	31	97

[Description] Return:
Status, 4 bytes contained the UID.

[Notes]
[Default]
[Reference]
[Example]

$1F $62

Devices: *all printer models with RFID reader/writer*

[Name] **Mifare Authentication, keys EEPROM (Mf1K, Mf4K)**

[Format]

ASCII	US	b	t	s	u0…u3 b
Hex	1F	62	t	s	u0…u3 b
Decimal	31	98	t	s	u0…u3 b

[Description] Authentication with t type key contained in the sector s of the EEPROM of PCD b block of TAG with UID u0…u3.

t = Type of key
t = $41 Authentication with A type key
t = $42 Authentication with B type key

0 ≤ s ≤ 15 = number of sector containing the key used for authentication.
0 ≤ b ≤ 63 = number of block to authenticate.

u0…u3 = 4 bytes containing the UID of TAG to authenticate.

Return:
status

[Notes]
[Default]
[Reference]
[Example]

$1F $63

Devices: *all printer models with RFID reader/writer*

[Name] **ISO 14443-A Anticollision (Mf1K, Mf4K, MfUL)**

[Format]

ASCII	US	c	n	p
Hex	1F	63	n	p
Decimal	31	99	n	p

[Description] Returns the complete UID (4 bytes) of one of the TAG present in RF.
n = collision level

n = $93 standard select code
n = $95 cascaded level 1
n = $97 cascaded level 2

p = bit number known in the UID
0≤ p ≤ 32

Return:
Status, UID0,UID1,UID2,UID3.

[Notes]
[Default]
[Reference]
[Example]

$1F $64

Devices: *all printer models with RFID reader/writer*

[Name] **ISO 14443 A Select (Mf1K, Mf4K, MfUL)**

[Format]

ASCII	US	d	n	u0…u3
Hex	1F	64	n	u0…u3
Decimal	31	100	n	u0…u3

[Description] Select the specified UID level

n = $93 standard select code
n = $95 cascaded level 1
n = $97 cascaded level 2

u0...u3 = selected UID

Return:
Status

[Notes]
[Default]
[Reference]
[Example]

$1F $68

Devices: *all printer models with RFID reader/writer*

[Name] **Mifare Halt Card (Mf1K, Mf4K, MfUL)**

[Format]

ASCI	US	h
Hex	1F	68
Decimal	31	104

[Description] Set with the Halt status the previous selected paper.

Return:
Status

[Notes]
[Default]
[Reference]
[Example]

$1F $6B

Devices: *all printer models with RFID reader/writer*

[Name] **Mifare Authentication with Keys (Mf1K, Mf4K)**

[Format]

ASCII	US	k	t	u0…u3	b	k0…k5
Hex	1F	6B	t	u0…u3	b	k0…k5
Decimal	31	107	t	u0…u3	b	k0…k5

[Description] Authentication with t type key write in k0…k5 b block of TAG with UID u0…u3.

t = $41 Authentication with A type key
t = $42 Authentication with B type key

0 ≤ b ≤ 63 = number of block to authenticate.

u0…u3 = 4 bytes containing the UID of TAG to authenticate.

K0…k5 = 6 bytes of the decrypted key.

Return:
status

[Notes]
[Default]
[Reference]
[Example]

$1F $71

Devices: *all printer models with RFID reader/writer*

[Name] **Mifare Request (Mf1K, Mf4K, MfUL)**

[Format]

ASCI	US	q	n
Hex	1F	71	n
Decimal	31	113	n

[Description] Execute a request at the TAGs present:
n = $52 All the TAG present answer to request, also the TAG in HALT status.
n = $26 Only the TAG in IDLE status answer to request.

Return:
Status, ATQ_H, ATQ_L
Where ATQ is (Answer to request) and e identify the type of TAG.

[Notes]
[Default]
[Reference]
[Example]

$1F $72

Devices: *all printer models with RFID reader/writer*

[Name] **Mifare Read(Mf1K, Mf4K, MfUL)**

[Format]

ASCI	US	r	n
Hex	1F	72	n
Decimal	31	114	n

[Description] Read the 16 bytes of the n sector specified.

0 ≤ n ≤ 63 * Depends on the type of TAG

Return:
Status, d0…d15.

[Notes]
[Default]
[Reference]
[Example]

$1F $73

Devices: *all printer models with RFID reader/writer*

[Name] **Mifare Select (Mf1K, Mf4K, MfUL)**

[Format]

ASCI	US	s	u0…u3
Hex	1F	73	u0…u3
Decimal	31	115	u0…u3

[Description] Select the TAG identified from the UID u0…u3

Returns:
Status

[Notes]
[Default]
[Reference]
[Example]

$1F $75

Devices: *all printer models with RFID reader/writer*

[Name] **Mifare ultralight multiple Write (MfUL)**

[Format]

ASCI	US	u	n	p	d0…dp
Hex	1F	75	n	p	d0…dp
Decimal	31	117	n	p	d0…dp

[Description] Write a number of byte equal to p on a Mifare Ultraligh type TAG starting from the block number n.

0 ≤ n ≤ 63 * Depends on the type of TAG

Returns:
Status

[Notes]
[Default]
[Reference]
[Example]

$1F $76

Devices: *all printer models with RFID reader/writer*

[Name] **Mifare ultralight multiple Read (MfUL)**

[Format]

ASCI	US	v	
Hex	1F	76	
Decimal	31	118	

[Description] Read the values of the 64 bytes of the TAG Ultralight.

Returns:
Status , d0…d64.

[Notes]
[Default]
[Reference]
[Example]

$1F $77

Devices: *all printer models with RFID reader/writer*

[Name] **Mifare ultralight multiple Write (MfUL)**

[Format]

ASCI	US	w	n	d0…d15
Hex	1F	77	n	d0…d15
Decimal	31	119	n	d0…d15

[Description] Write 16 bytes in the n block of the selected Mifare Ultraligh type TAGS.

0 ≤ n ≤ 63 * Depends on the type of TAG

Returns:
Status

[Notes]
[Default]
[Reference]
[Example]

2.2 List of possible Status values

MI_OK	(0)	MI_SENDBYTENR	(-51)
MI_CHK_OK	(0)	MI_CASCLEVEX	(-52)
MI_CRC_ZERO	(0)	MI_SENDBUF_OVERFLOW	(-53)
I1_OK	(0)	MI_BAUDRATE_NOT_SUPPORTED	(-54)
I1_NO_ERR	(0)	MI_SAME_BAUDRATE_REQUIRED	(-55)
MI_NOTAGERR	(-1)	MI_WRONG_PARAMETER_VALUE	(-60)
MI_CHK_FAILED	(-1)	I1_WRONGPARAM	(-61)
MI_CRCERR	(-2)	I1_NYIMPLEMENTED	(-62)
MI_CHK_COMPERR	(-2)	I1_TSREADY	(-63)
MI_EMPTY	(-3)	I1_TIMEOUT	(-70)
MI_AUTHERR	(-4)	I1_NOWRITE	(-71)
MI_PARITYERR	(-5)	I1_NOHALT	(-72)
MI_CODEERR	(-6)	I1_MISS_ANTICOLL	(-73)
MI_SERNRERR	(-8)	I1_COMM_ABORT	(-82)
MI_KEYERR	(-9)	MI_BREAK	(-99)
MI_NOTAUTHERR	(-10)	MI_NY_IMPLEMENTED	(-100)
MI_BITCOUNTERR	(-11)	MI_NO_MFRC	(-101)
MI_BYTECOUNTERR	(-12)	MI_MFRC_NOTAUTH	(-102)
MI_IDLE	(-13)	MI_WRONG_DES_MODE	(-103)
MI_TRANSERR	(-14)	MI_HOST_AUTH_FAILED	(-104)
MI_WRITEERR	(-15)	MI_WRONG_LOAD_MODE	(-106)
MI_INCRERR	(-16)	MI_WRONG_DESKEY	(-107)
MI_DECRERR	(-17)	MI_MKLOAD_FAILED	(-108)
MI_READERR	(-18)	MI_FIFOERR	(-109)
MI_OVFLERR	(-19)	MI_WRONG_ADDR	(-110)
MI_POLLING	(-20)	MI_DESKEYLOAD_FAILED	(-111)
MI_FRAMINGERR	(-21)	MI_RECBUF_OVERFLOW	(-112)
MI_ACCESSERR	(-22)	MI_WRONG_SEL_CNT	(-114)
MI_UNKNOWN_COMMAND	(-23)	MI_WRONG_TEST_MODE	(-117)
MI_COLLERR	(-24)	MI_TEST_FAILED	(-118)
MI_RESETERR	(-25)	MI_TOC_ERROR	(-119)
MI_INITERR	(-25)	MI_COMM_ABORT	(-120)
MI_INTERFACEERR	(-26)	MI_INVALID_BASE	(-121)
MI_ACCESSTIMEOUT	(-27)	MI_MFRC_RESET	(-122)
MI_NOBITWISEANTICOLL	(-28)	MI_WRONG_VALUE	(-123)
MI_QUIT	(-30)	MI_VALERR	(-124)
MI_CODINGERR	(-31)		

3 HF ASK COMMANDS

Introduction

The GEN5XX family of coupler is compliant with ISO 14443 A/B and Felica (ISO 18092) norms. It can handle all existing cards in the ASK range (CD97, GTML, CT2000, Mifare, contactless tickets CTS, CTM, ePassport and eID) and all ISO 14443-4 A/B and FELICA compliant cards and tickets (see below for full list of supported cards). The NFC items (Card and Cell-phones for example) are also supported but only in passive mode.

It supports 3 types of modulation specified by the norms (ISO14443 A and B and Felica) and is able to support the ISO14443/3 and 4 communication protocols.

This coupler is made up of a software module (also called CSC) which handles both low layers of the card communication (contact and contactless) and security layer by way of a contact security Application Module (SAM), as well as a global application layer dealing with successions of card/SAM commands enabling rapid and secure transactions without contact.

The user of the GEN5XX coupler may also implement many different supported cards/ticket, through high-level commands, without bothering about details of the card/SAM commands set.

The GEN5XX coupler has been designed to keep maximum compatibility with previous version of ASK coupler families such as GEN3XX and GEN4XX coupler.

The main differences therefore concern:

- RF treatment (15693 is not implemented on all version but the Felica® is now managed)
- // interface is not implemented (anyway, it still can be implemented on demand).

The aim of this document is to present the software interface of this coupler:

- General description of operation
- Communication Protocol
- Description of communication scenarios
- Details of commands

Applications

- Transport ticketing
- Payment
- Access control
- E-Paper

Functions supported

- System functions, for example configuration of the coupler
- Software download functions
- Card functions, specific to each type of card handled

Cards supported

- CD97 RJJ and RJL masks
- GTML, GTML2, CT2000
- CTS 256 and 512B Ticket
- CTM 521B Ticket
- Mifare® Standard 1K, Mifare® 4K, Mifare® UltraLight
- Mifare® ProX®
- Philips SmartMX® and DESFire®
- Motorola MV5000
- ePassport and eID
- ST ST19WR66
- ST ST19XR34
- Sharp
- Atmel AT90SC
- Any ISO 14443-4 A/B compliant card and ticket
- Any Felica® and NFC Passive devices
- ST SR Family

3.1 GENERAL OPERATION

3.1.1 Introduction

The GEN5XX reader communicates through a serial or a USB link and behaves as slave during all exchanges it may have with the host (the latter being an application card, a computer, etc.).

3.1.2 Description of interface

The communication interface with the host computer can be USB, RS232C, RS485 (for electrical interface), or TTL serial.
The physical characteristics of this interface, as well as the mode of selection are described in the hardware specification documents for GEN5XX couplers.
The rate of serial interface is programmable. The default value retained is 115 200 bauds.
This default value may be changed by the user using a system command.
Rate is programmable between 9600 baud and 691200 baud's.
- The link format in transmission and reception is: 8 bits, no parity, 1 stop bit

3.2 COMMUNICATION PROTOCOL

3.2.1 Description

The reader behaves as slave, thus waits for a command coming from the host (master), carries out this command and sends to the host the response to this command.
As soon as the reader is switched on, the host must before anything else, carry out a zero reset of the reader.
After this RESET of the reader, the host may send all the commands necessary to run the application.

3.2.2 Reader RESET Procedure

The reader is re-initialized at each power-up.
The first command after initialization must be the SoftwareVersion Command which allows the use of other commands.

3.2.3 Frame format

There are two types of message: 'pure command' messages (reduced to CMD byte) and the messages containing an application command (system and card commands).

3.2.3.1 Command frames

CMD (1 byte)	See ‘Detail of CMD and STA registers'
LNG (1 or 2 byte(s))	See ‘Frame length’
DATA (x byte)	See ‘Details of reader functions’ See 'Software interface'
End of frame = $00	Fixed value 0x00
CRC (2 bytes) CRCL CRCH	CRC CCITT of all the previous bytes See ‘Calculation of CRC’

The host sends this frame, and the bytes may be sent one after another without interruption. The maximum time between each byte sent by the host is checked by timer (around 1500 ms).
The ‘LNG’ value is the length of the ‘DATA’ block in the frame.
The ‘End of frame' byte is a byte which means to the reader that the ‘DATA’ block is finished and that the two following blocks are the CRC.
The protocol does not include a mechanism of resumption on communication error.

3.2.3.2 Response frame

STA (1 byte)	See ‘Detail of CMD and STA registers'
LNG (1 or 2 byte(s))	See ‘Frame length’
DATA (x byte)	See ‘Details of reader functions’ See 'Software interface'
End of frame = $00	Fixed value 0x00
CRC (2 bytes) CRCL CRCH	CRC CCITT of all the previous bytes See ‘Calculation of CRC’

This frame is sent by the reader, in response to a command sent by the host. All the bytes are sent one after the other without interruption.
The ‘LNG’ value is the length of the ‘DATA’ block in the frame.
The ‘End of frame' byte is a byte which means to the reader that the ‘DATA’ block is finished and that the two following blocks are the CRC.

3.2.3.3 Description of pure commands

CMD (1 byte)	See ‘Detail of CMD and STS bytes'

It concerns commands reduced to CMD and whose bit 7 (EXEC) is at zero.
CMD = STOP that is to say $02
CMD = RST that is to say $01
Only this byte is sent by the host.
(Remark: the only command that may be interrupted by stop is Enter_HuntPhase)

3.2.3.4 Description of responses to pure commands

CMD (1 byte)	See ‘Detail of CMD and STA bytes'

It concerns the response to a pure command; this response is reduced to a single ‘STA’ byte
After a pure command CMD = STOP
STA = ABORT that is to say $04
After a pure command CMD = RST
STA = RES that is to say $10

3.2.4 Details of CMD and STA bytes

	7	6						0
CMD	EXEC	EXT					STOP	RES

Note: The bits are active at 1 and the unused bits must be fixed at 0. The bits EXEC, STOP and RES are Exclusive Operation and may not be used simultaneously.

RES... Reset
Complete Initialization of CSC.

STOP... Stop
Cancels current command.
This command only acts on interrupting a current Enter Hunt Phase command and in this case the reader sends back "STA = ABORT"; otherwise the reader is mute.

EXEC... Execute
Executes the command transmitted by the host into the following bytes.

EXT... Extended
CMD byte is followed by LNG LOW and LNG HIGH.

	7	6						0
STA	ERR	EXT		RES		ABORT		DATA

Note: The bits are active at 1.

DATA... Data available
Data is transmitted following a command received.

ABORT... Abort execution
The interruptible current command was stopped.

RES... Reset
The CSC was re-initialized by a physical or logical RESET (CMD.RES).
Indicates in addition that auto-check succeeded.

ERR... Error
The command syntax is erroneous (class, function unknown).

EXT... Extended
CMD byte is followed by LNG LOW and LNG HIGH.

3.2.5 CRC Calculation

The CRC concerns all characters in the frame, including CMD and STA and 0. It is the CRC on 16 bits defined by the norms ISO 3309, CCITT V42 and CCITT X25. A portable implementation in C, usable without modification on PC for calculation and verification of CRC, is available (ref. CSC_ORD.C).
Examples: This function enables calculation of the CRC of bytes in the frame which are in the buffer of non-signed characters 'FRAME'. The length of bytes in this frame are found in 'LNG'

```
int LNG;
unsigned char TRAME[256];
void SetCRC(void)
{
unsigned short CRCVal=0; int i ;
for( i = 0 ; i < LNG ; i++ )
CRCVal = TABLE[( CRCVal ^ = TRAME[ i ] ) & 0xFF ] ^ ( CRCVal >> 8 );
TRAME [ LNG ] = CRCVal % 256;
TRAME [ LNG + 1 ] = CRCVal / 256;
LNG = LNG + 2;
}
```

In 'TABLE' may be found the calculation constants of the CRC

```
const unsigned short TABLE[ 256 ]={
       0xF078,0xE1F1,0xD36A,0xC2E3,0xB65C,0xA7D5,0x954E,0x84C7,0x7C30,0x6DB9,0x5F22,0x4EAB,
       0x3A14,0x2B9D,0x1906,0x088F,0xE0F9,0xF170,0xC3EB,0xD262,0xA6DD,0xB754,0x85CF,0x9446,
       0x6CB1,0x7D38,0x4FA3,0x5E2A,0x2A95,0x3B1C,0x0987,0x180E,0xD17A,0xC0F3,0xF268,0xE3E1,
       0x975E,0x86D7,0xB44C,0xA5C5,0x5D32,0x4CBB,0x7E20,0x6FA9,0x1B16,0x0A9F,0x3804,0x298D,
       0xC1FB,0xD072,0xE2E9,0xF360,0x87DF,0x9656,0xA4CD,0xB544,0x4DB3,0x5C3A,0x6EA1,0x7F28,
       0x0B97,0x1A1E,0x2885,0x390C,0xB27C,0xA3F5,0x916E,0x80E7,0xF458,0xE5D1,0xD74A,0xC6C3,
       0x3E34,0x2FBD,0x1D26,0x0CAF,0x7810,0x6999,0x5B02,0x4A8B,0xA2FD,0xB374,0x81EF,0x9066,
       0xE4D9,0xF550,0xC7CB,0xD642,0x2EB5,0x3F3C,0x0DA7,0x1C2E,0x6891,0x7918,0x4B83,0x5A0A,
       0x937E,0x82F7,0xB06C,0xA1E5,0xD55A,0xC4D3,0xF648,0xE7C1,0x1F36,0x0EBF,0x3C24,0x2DAD,
       0x5912,0x489B,0x7A00,0x6B89,0x83FF,0x9276,0xA0ED,0xB164,0xC5DB,0xD452,0xE6C9,0xF740,
       0x0FB7,0x1E3E,0x2CA5,0x3D2C,0x4993,0x581A,0x6A81,0x7B08,0x7470,0x65F9,0x5762,0x46EB,
       0x3254,0x23DD,0x1146,0x00CF,0xF838,0xE9B1,0xDB2A,0xCAA3,0xBE1C,0xAF95,0x9D0E,0x8C87,
       0x64F1,0x7578,0x47E3,0x566A,0x22D5,0x335C,0x01C7,0x104E,0xE8B9,0xF930,0xCBAB,0xDA22,
       0xAE9D,0xBF14,0x8D8F,0x9C06,0x5572,0x44FB,0x7660,0x67E9,0x1356,0x02DF,0x3044,0x21CD,
       0xD93A,0xC8B3,0xFA28,0xEBA1,0x9F1E,0x8E97,0xBC0C,0xAD85,0x45F3,0x547A,0x66E1,0x7768,
       0x03D7,0x125E,0x20C5,0x314C,0xC9BB,0xD832,0xEAA9,0xFB20,0x8F9F,0x9E16,0xAC8D,0xBD04,
       0x3674,0x27FD,0x1566,0x04EF,0x7050,0x61D9,0x5342,0x42CB,0xBA3C,0xABB5,0x992E,0x88A7,
       0xFC18,0xED91,0xDF0A,0xCE83,0x26F5,0x377C,0x05E7,0x146E,0x60D1,0x7158,0x43C3,0x524A,
       0xAABD,0xBB34,0x89AF,0x9826,0xEC99,0xFD10,0xCF8B,0xDE02,0x1776,0x06FF,0x3464,0x25ED,
       0x5152,0x40DB,0x7240,0x63C9,0x9B3E,0x8AB7,0xB82C,0xA9A5,0xDD1A,0xCC93,0xFE08,0xEF81,
       0x07F7,0x167E,0x24E5,0x356C,0x41D3,0x505A,0x62C1,0x7348,0x8BBF,0x9A36,0xA8AD,0xB924,
       0xCD9B,0xDC12,0xEE89,0xFF00};
```

3.2.5.1 CRC format

The frames only have a CRC if they are not reduced to a single byte (pure command).The first byte after the 'End of frame' byte is the lower part of CRC (LSB)The last byte in the frame is the upper part of the CRC (MSB).

3.2.6. Frame length

Normal mode (EXT bit =0 in CMD or STA byte)

The maximum number of bytes transmitted with the card is a bit more than 256. That's why the length indicator 'LEN' may exceed the 0xFF hexadecimal value. In the case of a length greater than or equal to 255, the length will be encoded on two bytes: the first one LEN1 will be set to 0xFF, and the second one LEN2 will represent the rest until the total length, so as LEN will be equal to LEN1 + LEN2. Hence, until 0xFE, LEN will remain on one byte, and beyond this value will be on two bytes:

254 → 0xFE
255 → 0xFF 0x00
256 → 0xFF 0x01, and so on…

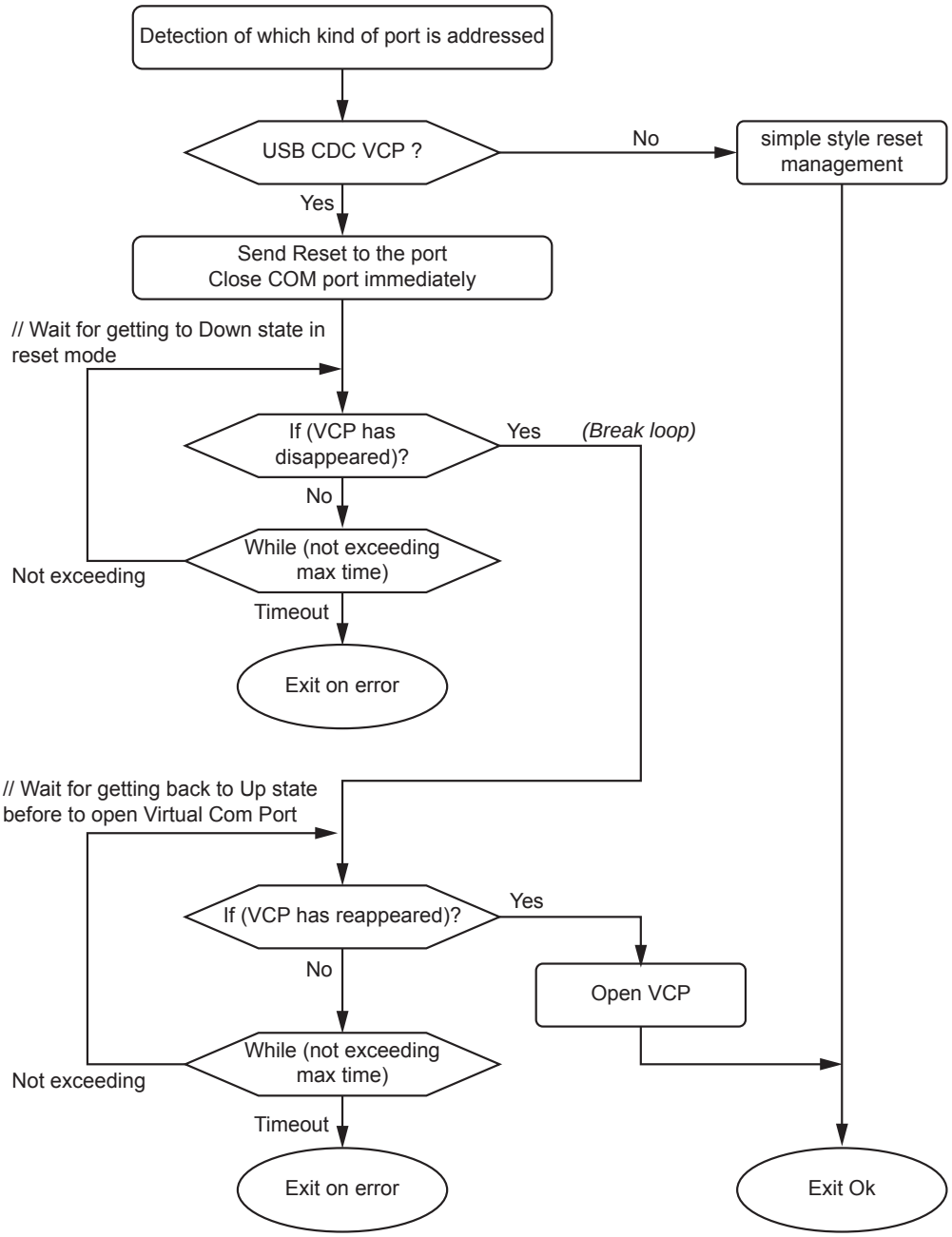
Extended mode (EXT bit =1 in CMD or STA byte)

The frame length is 2 bytes long: LNG LOW and LNG HIGH.
The effective frame length is LNG LOW + (256 * LNG HIGH)
The maximum frame length is 800 bytes.

3.2.7 Note on the USB interface

There is a difference with GEN4xx family and GEN5XX family on the USB interface management.
The USB interface of the GEN5xx family of coupler is based on the USB Class CDC model. It means that a virtual serial port is created to communicate with the coupler.
As it is based on the microcontroller a difference appear in the reset management.
From one side, the microcontroller cannot keep the USB link up during the reset phase.
On the other side, due to Operating System management, it is necessary to close the com port on the host side immediately after sending the reset command (or equivalent) in order to avoid dead lock when resetting the coupler. Otherwise operating system will consider a session still being opened with this port and will not be able to retrieve a good communication mode after wise.
This particular case is taken into account in the last version of library ASKCSC.DLL which manage correctly all version of ASK coupler communication.
In order to help developers who has to implement their own interface, sources of the DLL are given with the development kit and a set of function is given in a dedicated file to help for the management of USB VCP or other serial port.

Algorithm should then be as follow:



3.3 DESCRIPTION OF COMMUNICATION SCENARIOS

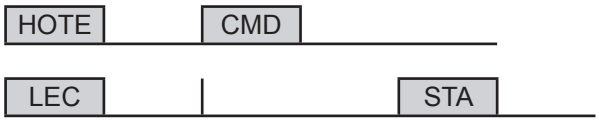
3.3.1 Restriction on the Master/slave model

The computer should specify a maximum time for the polling command (3 seconds for example), and on that time is reached it can still interrupt the polling through the STOP command.
The RESET command can be send at any time.

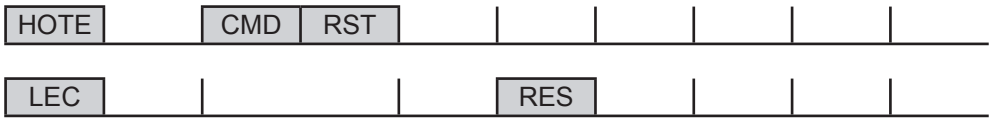
3.3.2 Frames

3.3.2.1 Pure command

This exchange is carried out on a RESET or on a STOP command; the host sends only a single byte.
(Remark: the only command that may be interrupted by stop is the Enter_HuntPhase polling)
The reader sends back a single byte of 'STA' status in the event of command acknowledgement.

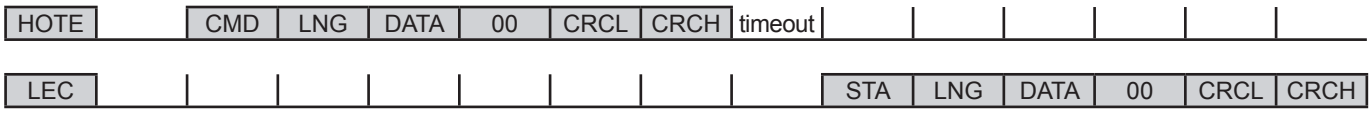


In the event that the reader does not recognize the command before xx ms (xx depends on estimated time of command execution), the host can carry out a RESET of the reader.

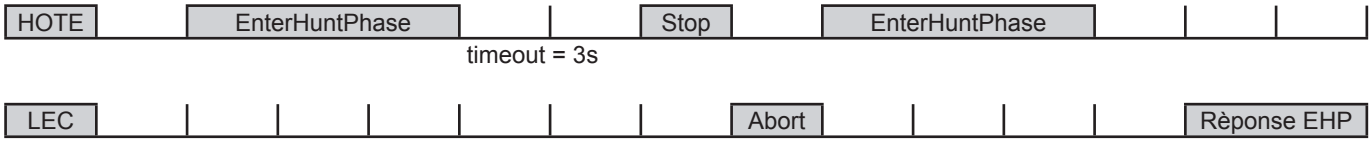


3.3.2.2 Standard command

The timeout between issuing the command and the reader's response depends on the command sent.



The old style EnterHuntPhase commands do not call for responses (search for badge/card) if no card is detected. Anyway, it is possible to use the parameter "timeout" of the command and use it periodically. However, for compatibility reasons the STOP command is kept.



Stop: a pure command CMD = STOP ($02)
Abort: the response to this pure command STA = ABORT ($04)
EnterHuntPhase: issue of CMD+LNG+DATA+ $00 + CRC
That is to say $80 $07 $01 $03 $00 $00 $00 $00 $01 $00 $65 $18 (See 'Details of reader functions')
EHP answer: the response to EnterHuntPhase if a card is found by the antenna.

3.4 DESCRIPTION OF THE CONTACT CARD INTERFACE

3.4.1 Description

Remark on the Contact Interface:
The Contact interface is a set of up to 5 slots which can be configured in the Innovatron SAM Speed protocol (up to 423 750 bauds) or the ISO 7816 Bauds (some baud rates are not fully supported).
When the Contact interface is selected in the EnterHuntPhase parameters, if a device is found in the fifth Slot it is considered as a contact card and it is used in the ISO7816 protocol.

3.4.2 Restriction

The Limitation on the implementation of the 7816 protocol is as follow:
- Only 5 Volt interface restriction : Type A or A&B (no support of Type B card)
- VPP is not connected (thus if such a card is inserted in the slot, a warning will occurs in the status of the connection)
- No clock suspends procedure implemented.
- Limitation to 4 levels of interface character in the ATR response interpretation
- Specific Fixed Guard Time implementation != TC(1) (1 second)
- Transparent mode with automatic GetResponse implementation.

N.B.: The T=1 protocol is not implemented yet (in the current version).

3.4.3 Status Description

Returned Contact Status :
0x00: Correct, No Warning.
0x82: Correct, Warning TA1 value will not be taken into account because no PPS procedure is available
0x81: Correct, Warning TB1 indicates that the Writing condition could not be met.
0xF0: Bad Ack Answer
0xF1: Bad SWISW2 Answer
0xF3: Character Sending Error
0xF4: Character Receiving Error
0xF5: PPS Failed
0xF6: Bad TA1 parameter
0xF7: Bad TP parameter
0xF8: Bad TCK control
0xF9: Bad PPS Mode
0xFA: SAM Signal Locked
0xFB: SAM Timeout
0xFC: SAM Length Error
0xFD: SAM Not defined
0xFE: incorrect Parameter
0xFF: Timeout occurs during dialogue or SAM not detected.

3.5 DESCRIPTION OF THE NEW ENTER HUNT PHASE FEATURES

Default configuration for Card Search is
- Max Number of cards to look up in ISO 14443-B mode is 1
- Type Of ReqB (ReqB or WakeUp) is 00 (REQB)
- Number of slots in case of collision is 00 (No slot marker : only probabilistic method has been tested)
- AFI value to seek is 00 (All)
- Automatic Select Diversifier implementation is Yes for the Normal mode and no for the RS485 mode (acts on both Innovatron and ISO14443B card)
- Deselecting of the cards by just switching the field off (value = 0).
- SelectApplication() is sent at the end of the EHP procedure, for the ISOB cards (value = 1).

All theses values can be changed by the Class 01 command 17: F01_17_REQ_PARAM
Parameters in :
- Byte 0: Max Number of cards to look up at (1 up to 5)
- Byte 1: Type Of ReqB (ReqB =0 or WakeUp = 1)
- Byte 2: Number of slots in case of collision (0 = No Slot marker, 1 to 14 otherwise)
- Byte 3: AFI value to look for: (see the ISO 14443 norm)
- Byte 4: Automatic Selection of Diversifier: (0 to disable automatic)
- Byte 5: Real deselection of the cards (if 1) or only Field On/Off (if 0).
- Byte 6: The SelectApplication() command will be sent at the end of the EHP for ISOB cards(if 1).

If the following command is sent: 01 17 00, the complete status of all the parameters in their current value will be given back.

The choice « No automatic selection of the Select Diversifier » is useful for:
- For the multiSam use to avoid sending this command to a SAM before that the application knows which card is detected and so can choose the good SAM slot.
- For the multiword detection to avoid sending this command to the SAM before knowing which of the card will be chosen first.
- For the RS485 to avoid lacks of communication affected by the critical section of the SAM communication routine.

To allow this choice a new command to Select Diversifier is implemented
Function: F01_16_Select_DIV
Parameters in :
- Byte 0: Sam Slot
- Byte 1: protocol
- Byte 2-5: Card Serial Number

Returned value:
- Status of operation 1 = Ok, 0 = Nok

If the “Timeout” mode is chosen (mode = 1), the search will be performed for the all selected types until the timeout reaches 00 (the timeout value to be written in the EnterHuntPhase command is in 10ms unit). The order of the search is as follows:
- GTML/CD97
- Ticket
- Mifare
- ISO A
- ISO B
- FELICA
- MV5000
- Contact

When the timeout is reached, thanks to a raised flag, the search is given up. Otherwise, the following type of card will be looked for. At the end of the loop, if the timeout is still positive, we go on at the beginning of the searching loop.

3.6 SOFTWARE INTERFACE

This chapter describes the structures of data sent and received on the DATA zones defined in chapter 2.4. Frame.

3.6.1 Classes of commands

The software interface of the ASK reader is organized into classes of commands defining major types of functions:

- System commands: configuration, transparent commands "TAG" and "SAM"
- Macro-commands for cards (CD97, GTML, CT2000 etc....)

3.6.2 Structure of commands

The commands are defined by:

- Class (system, badge, etc.)
- Instruction (function number in the class)
- Associated data

CLASS 1 byte	INS 1 byte	Associated data xx bytes (270 bytes MAX)

This information is sent by the host to the 'DATA' zone of command for frames.
Details on available commands are provided in chapter 5 of this document.

3.6.3 Responses to commands

The responses to commands are identical to the 'DATA' zone of commands seen above.

3.6.4 Error reports

There are 3 levels of error reports:

- Error detected by the reader at the level of access interface to functions: class, unknown function or syntax error in system class command. These errors are reported by the reader to the host via the 'ERR' bit in STA register. There is no other report.
- Error detected by a reader function called by the host (class other than system class): the error is reported by the reader to the host by an execution report on 3 bytes: 1 byte reporting the origin of the error, 2 bytes of detail on the error.
- Error detected by the badge or a SAM: the error is reported by the reader to the host through an execution report on 3 bytes: 1 byte representing the error family, 2 bytes coming from the badge or the error details SAM.

The error report codes are proper to each function class.

3.6.5 Specific SAM Errors

The transparent command 01_14_SendToSAM communicates directly with the SAM and can therefore give back very specific error codes about the exchange with the SAM:

SAM_EXEC_OK	$00
SAM_BAD_ACKANSWER	$E1
SAM_BAD_SW12ANSWER	$E2
SAM_NR2S_ERROR	$E3
SAM_NR2R_ERROR	$E4
SAM_BAD_PPSEXEC	$E5
SAM_BAD_TA1PARAM	$E6
SAM_BAD_TPARAM	$E7
SAM_BAD_TCK	$E8
SAM_BAD_PPSMODE	$E9
SAM_SIGNAL_LOCKED	$EA
SAM_TIME_OUT	$EB
SAM_LENGTH_ERROR	$EC
SAM_NOT_DEFINED	$ED
SAM_INCORRECT_PARAM	$EE
SAM_NOT_DETECTED	$FF

3.7 DETAILS OF ASK READER FUNCTIONS

3.7.1 DOWNLOAD Class (N°= $00)

$00 $01

Devices: *all printer models with ASK module*

Name: Download start
Description : Starts software download.

CLASS	INS	DATA IN
$00	$01	-

No associated input data

CLASS	INS	DATA OUT
$00	$01	Status(1)

STATUS : 1 byte
0x00 No error
0x01 Error while erasing sector 1
0x02 Error while erasing sector 2
0x03 Error while erasing sector 3
0x0B & 0x0E Error in programming the Flag in EEPROM before loading
0x0C Error in programming the Flag in EEPROM after loading

$00 $04

Devices: *all printer models with ASK module*

Name: Change of default rate
Description : Changes the rate of a series link in RS232/RS485/TTL series (default rate configured in factory at 115200 baud). The new value will be taken into account at next reset of the coupler.

CLASS	INS	DATA IN	DATA IN	DATA IN
$00	$04	RS232 (1)	RS485 (1)	TTL (1)

RS232 : baud rate divider on RS232 serial link. (Baud rate divider is 1382400 / BAUDRATE)
RS485 : baud rate divider on RS485 serial link. (Baud rate divider is 1382400 / BAUDRATE)
TTL : baud rate divider on TTL serial link. (Baud rate divider is 1382400 / BAUDRATE)

CLASS	INS	DATA OUT
$00	$04	Status(1)

STATUS : 1 byte : 0x00 Failure
0x01 Success

$00 $06

Devices: *all printer models with ASK module*

Name: Write SAM Number

Description : Writes in the EEPROM the SAM Number to use by default. This value will be useful for the initialization of the CSC, when selecting the SAM to use at the beginning, before the first transactions.

CLASS	INS	DATA IN
$00	**$06**	**Number**

Number : number of the SAM to use by default.

Note: if a value below 1 or above 4 is written, the default SAM will be the SAM number 1.

CLASS	INS	DATA OUT
$00	**$06**	**Status(1)**

STATUS : 1 byte : 0x00 Failure
0x01 Success

$00 $07

Devices: *all printer models with ASK module*

Name: Write Config Eeprom
Description : Writes in the EEPROM configuration.

CLASS	INS	DATA IN	
$00	**$07**	**Index**	**Value**

Index:
$01 : Value = serial (RS232/TTL/RS485) baud rate divider =1382400 / BAUDRATE
$02 : Not relevant for GEN5xx : kept for compatibility with other products
$03 : Not relevant for GEN5xx : kept for compatibility with other products
$04 : Value = default SAM Number
$05 : Field off CTx : turn on the field before CTx command and turn off the field during (Value * 1 ms) after CTx command. 0x00 and 0xFF disable field management on CTx
$06 : Auto Led management enabled if Value = 1. Leds are managed by firmware (red = power on, orange = field on, green = reader/card communication.
$07 : Not significant on GEN5xx
$08 : Host communication frame padding : module 62 byte padding if Value = 62.
$09 : ISO14443-4 number of retries.
$0A : Delay between retries (ms).
$0B : default RX RF speed at reset (00=106, 01=212, 02=424, 03=847 kb/s).
$0C : default RX RF speed at reset (00=106, 01=212, 02=424, 03=847 kb/s).
$0D : SAM reset at coupler reset (0=no reset)
$0E : AUX Pin signal
$0F : High baud rate ISO14443-A gain (00=20, 01=24, 02=31, 03=35 dB)
$10 : Last Slot switch test (1=yes (CAM), other = no (SAM))
$11 : Strict ISO14443-3B timeout (1=strict check, other = no strict check, same as GEN3XX)
$12 : Strict ISO14443-4B timeout (1=strict check, other = no strict check, same as GEN3XX)
$13 : Delay after REQ/Select (0 or FF : no delay, same as GEN3XX, other = delay in ms)
$14 : Unconditional Mifare selection before authentication (if value=1)
$15 : Not significant on GEN5XX
$16 : Custom Frame Waiting Time (Value * 10 ms, 00 or FF = no custom FWT)
$17 : ISO14443-4 retries on PICC timeout (if value=1)

CLASS	INS	DATA OUT
$00	**$07**	**Status(1)**

STATUS : 1 byte : 0x00 Failure
0x01 Success

$00 $08

Devices: *all printer models with ASK module*

Name: Read Config Eeprom
Description : Read the value at the Index EEPROM.

CLASS	INS	DATA IN
$00	$08	Index (1)

Index: EEPROM configuration index

CLASS	INS	DATA OUT	
$00	$08	Status(1)	Value(1)

STATUS : 1 byte : 0x00 Failure
0x01 Success

Value regarding index : see Write Config Eeprom
Default values:

CSC Configuration

Index	Label	Hex Value	Signification
1	RS232 Speed	FF	115200 bauds
2	RS485 Speed	FF	115200 bauds
3	TTL Speed	FF	115200 bauds
4	Default SAM	01	SAM n°1
5	Field off CTX	1E	During 30 ms
6	Auto LED	FF	Disabled
7	15693 modulation	FF	100%
8	Frame padding	FF	Disabled
9	ISO14443-4 retries	FF	10
10	ISO14443-4 retries delay	FF	10 ms
11	Default RX RF speed	FF	424 kb/s
12	Default TX RF speed	FF	424 kb/s
13	SAM reset at startup	FF	Yes
14	AUX Pin	FF	VEvalL
15	High baud rate ISO14443A gain	FF	35 dB
16	Contact Slot 4 switch test	FF	No (SAM)
17	ISO14443-3B timeout strict check	FF	No
18	ISO14443-4B timeout strict check	FF	No
19	ISO14443 delay after REQ/SELECT	FF	0 ms
20	Mifare selection before authentication	FF	No
21	MUX482 selection same as MUX382	FF	No
22	ISO14443-AB custom Frame Waiting Time	FF	No
23	ISO14443-4 retries on PICC timeout	FF	Yes

3.7.2 SYSTEM Class (N°= $01)

This class is dedicated to module resource administration which is not specific to a card mask (communications, configurations, inputs/outputs, update, etc...).

$01 $01

Devices: *all printer models with ASK module*

Name: Software version

Description : Sends back reader software version and allows enabling or disabling the CRC computation for communications exchanges between the HOST and the CSC. N.B. this command initializes CSC communication.

CLASS	INS	DATA IN
$01	$01	CRC ON/OFF

CRC ON/OFF: 0xFF: CRC not needed. After reset, Version command with CRC must be issued before using CRC disabling.
All other values: CRC needed for each exchange.
By default, the CRC is needed.
This parameter is not mandatory. If not present in the command, the default value will remain unchanged.
CAUTION: this parameter has been implemented to increase the communication speed!

CLASS	INS	DATA OUT
$01	$01	VERSION (x)

VERSION : character chain of software version (terminated by $00).
GEN5XX CSC xx.yy<iii> Mmm jj yyy HH:MM:SS (C) ASK SAMs
Where
xx = version number
yy = revision number
iii interface USB, or serial baud rate from 9600 to 691 200 baud
Mmm = Month (3 first letters in ASCII), jj = Day, yyyy = Year
HH = Hour, MM = Minutes, SS = Seconds
s = SAM used by default (value written in EEPROM), equal to “?” if none selected.

$01 $03

Devices: *all printer models with ASK module*

Name: Enter Hunt Phase

Description : This function enables badges to be searched for in different modes and different ways. The command may be cancelled by a STOP command if no timeout is specified.

CLASS	INS	DATA IN												
$01	$03	MONO	ANT	OTH	NFC/ Cont	MV/ Felica	ISO B	ISO A	Mifare	Ticket	INNO	MODE	FORGET	TIMOUT

MONO :	1 Nibble : $4 for a single-shot search, $0 otherwise.	
ANT :	1 Nibble : kept for compatibility but unused.	
OTH :	1 Nibble : Other chip detection : at this time only SRI is supported.	
CONT :	1st half Nibble: (Reserved for future use).	
	Bit 1 : RFU	
	Bit 0 : ISO7816 contact Card search, available if extension board is present	
MV5K/FELICA:	1st half nibble (Most Significant Bits) = Number of MV5K search,	
	2nd half nibble (Least Significant Bits) = Number of Felica® search.	
ISO B :	1 Nibble : Number of successive card search to perform in ISO B norm protocol.	
ISO A :	1 Nibble : Number of successive card search to perform in ISO A norm protocol and mandatory for the MIFARE® UltraLight.	
MIFARE® :	1 Nibble : Number of successive card search to perform in MIFARE® protocol.	
TICKET :	1 Nibble : Number of successive card search to perform in CTS / CTM protocol.	
INNO :	1 Nibble : Number of successive card search to perform in Innovatron RF protocol.	
MODE :	Type of Enter Hunt Phase	
		$00 : short (compatibility with the old one)
		$01 : long (take into account the 2 following bytes)
FORGET :	1 byte	$00 takes into account the serial number of the last badge presented.
		$01 Forgets the serial number of the last badge presented.
TIMOUT:	1 byte (N x 10 ms)	
	Polling duration of the enter hunt phase. Beyond this time the reader replies a status.	
	This avoids the reader staying indefinitely awaiting a badge.	
	If the value is nil, waiting time is infinite.	

NB: If the MONO mode of the byte 0 is selected with the LONG MODE (byte 5 = 1), either the Time Out lasts less than the sequence of all the searches selected and the hunt stops before the whole sequence is achieved, or if greater the hunt stops after the end of the sequence, before the Time Out reaches the end.

NB: on multi-protocol cards (INNOVATRON and ISOB), ISOB nibble should be more than 1.

CLASS	INS	DATA OUT				
$01	$03	CNT (1)	COM (1)	LNG (1)	ATR (1)	... ATR (N)

CNT : 1 byte, kept for compatibility,
8x indicates that antenna x has broken down,
00 otherwise

COM : 1 byte, communication mode
$01 : Card recognized with Felica® protocol
$02 : Card recognized with ISO 14443 type A protocol (ISO level 4 compliant but not Calypso)
$03 : Card recognized with INNOVATRON protocol
$04 : Card recognized with ISO 14443 type B protocol (Calypso Card)
$14 : ISO 14443 type B protocol asked but an unwanted collision occurred.
$05 : Card recognized with ISO 14443 type MIFARE protocol
$15 : ISO 14443 type MIFARE protocol asked but an unwanted collision occurred.
$06 : CTS or CTM Ticket recognized
$07: Card recognized in contact mode
$08: Card recognized with ISO 14443 type A part 3 but not compliant with 14443 type A part 4
$18 : ISO 14443 type A protocol asked but an unwanted collision occurred.
$09 : Card recognized with ISO 14443 type B protocol (Non-Calypso card)
$0B : Card recognized with MV5000 protocol (no data format control)
$1B : MV5000 protocol asked but an unwanted collision occurred.
$0C : Card recognized with ISO 14443 type A protocol ((Calypso Card)
$0D : SRI detected
$0E : NFC Felica detected
$0F : NFC Mifare detected
$6F : Timeout expired.
$7F: Response (before tag research)

NB: no collision code has been implemented for the INNOVATRON or CTx cards.

LNG : 1 byte, length of cards responses
ATR : N bytes :

- INNOVATRON protocol :
  Long REPGEN data in the case of Innovatron protocol cards composed of
  - The serial number (4 bytes) followed by 2 bytes
  - The Answer to reset (17 bytes for CD97 and GTML)
  - The Status words (2 bytes)

- ISO 14443 Type B protocol :
  - The communication channel number CID (1 Byte)
  - The Serial Number if the card is Calypso-Compliant, otherwise the first 4 bytes are filled at 00 and the 4 others are the PUPI (8 bytes).
  - 1 byte optional length (1 to 16) of application name (if bit 3 of byte 6 of EnterHuntPhaseParameter = 1)
  - n byte optional application name (if bit 3 of byte 6 of EnterHuntPhaseParameter = 1)
  - The Historical Bytes if the card is Calypso-Compliant, otherwise the last 3 are filled at 00(7bytes for GTML2) same order as in the SelectApplication command
  - The Status words is returned if the card is Calypso-Compliant, filled at 00 otherwise (2 bytes)

- MIFARE® protocol:
  The answer in this mode depends on two parameters: for the request to be really sent in MIFARE® mode, the presence of the MIFARE® chip in the reader is mandatory AND only one card must be looked for. Otherwise the answer is formatted just as an ISO 14443 Type A (14443-4 compliant or not). But if the search is a true MIFARE® search, the format is as follows:
  - Communication Status (1 Byte): set to 0x00: OK.
  - Type of the card (e.g.: 0x08 for the Mifare® Classic, 0x18 for the Mifare 4k, 0x28 for the Mifare Classic implementation in ProX) (1 Byte)
  - The 4 Serial Number bytes

- ISO 14443 Type A protocol :
  - ISO 14443-4 compliant:
    - The communication channel number CID (1 Byte).
    - Length of the Serial Number (1 Byte).
    - The Serial Number if the card is Calypso-Compliant, otherwise the UID (4, 7 or 10 bytes).
    - 1 byte optional length (1 to 16) of application name (if bit 3 of byte 6 of EnterHuntPhaseParameter = 1)
    - n byte optional application name (if bit 3 of byte 6 of EnterHuntPhaseParameter = 1)
    - Length of the Information (1 Byte).
    - Information given back in the ATS, including:
      - Maximum size of a frame accepted by the card. (1 Byte)
      Warning: 0xFF means 256 bytes!
      - The bit rate from the coupler to the card. (1 Byte)
      - The bit rate from the card to the coupler. (1 Byte)
      - Only the same baud rate for both direction is supported? -> 1 if TRUE. (1 Byte)
      - The Frame Waiting Time Integer which defines the Frame Waiting Time between the end of the frame sent by the coupler and the beginning of the answer of the card. (1 Byte)
      - The Specific Guard Time Integer which defines the time needed by the card before answering after sending the ATS. (1 Byte)
      - NAD supported? -> 1 if TRUE. (1 Byte)
      - CID supported? -> 1 if TRUE. (1 Byte)
      - Information of the Application (historical bytes).
  - ISO 14443-4 non-compliant: (for instance the MIFARE®UltraLight)
    - Set to 00 (1 Byte) (means non ISO–4 compliant).
    - Length of the following Serial Number (1 Byte).
    - Serial Number (Length Bytes)

- CTS or CTM Ticket recognized, 9 bytes (resp. 10) if the EnterHuntPhase has been processed in mode 1 (resp. mode 0) corresponding to:
  In mode 1 (long EnterHuntPhase):
  - Ticket status (1 byte) (0x0F if detection successful)
  - Product code / Manufacturer code (2 bytes)
  - Application code / Embedder code (2 bytes)
  - Serial number in (4 bytes)

  In mode 0 (short EnterHuntPhase):
  - Manufacturer code / Product code (2 bytes)
  - Embedder code / Application code (2 bytes)
  - Serial number in reverse order(4 bytes)
  - System and Invalidation bits (2 bytes)

- SRI recognized:

- Status (1 byte):	$00:	communication interrupted
	$01:	bad CRC
	$0F:	success
	$80:	collision
- Chip type (1 byte):	$00:	SR176
	$01:	SR512
	$02:	SR4K
- UID:	64-bit (8-byte) UID from LSB to MSB (UID0, UID1… UID7)	

- ISO7816 Contact mode: answer to reset, whose length depends on the different kinds of cards.
- Motorola cards :
  - MV5000 cards:
    - The DAD byte (1 Byte).
    - The PUPI (4 Bytes).
    - Length of the Historical bytes (1 Byte = 0x04).
    - Historical bytes
    - Length of the Serial Number (1 Byte = 0x18).
    - The Serial Number

- FELICA protocol :
  Answer to Req Felica data composed of
  - The response code (1 byte) == 0x01
  - The IDm number (8 bytes) : Card Unique Manufacture IDs
  - The PMm numver (8 bytes) : Manufacture Parameters :
    - IC Code : (2 bytes)
    - Request Service Command (1 Byte)
    - Request Response / Search Service Code / Request System Code Commands (1 Byte)
    - RFU (1 Byte)
    - Read Without Encryption Command (1 Byte)
    - Write Without Encryption Command (1 Byte)
    - RFU (1 Byte)
  - The System code (2 bytes) when required

The status words -> 0x90 XX where XX is the MF status
0xX0: card is valid
0xX1: card is invalided
0x0X: no PIN presentation error
0x1X: one PIN presentation error
0x3X: two PIN presentation error
0x7X: PIN blocked

Remark 1 : For detection of several ISO14443 cards, (if multi-detection is activated for the ISO card), the card response are appended several times in the response data field.
Remark 2 : For each new detection of ISO A or B by the EnterHuntPhase commands, the data for cards previously found are lost, so it is not possible to communicate with both type A and type B cards in the same time.
Remark 3: By default, the EnterHuntPhase command sends WAKE UP requests. So if multi-detection is wished, the selection of the REQ request must be previously done thanks to the 01_17_SetEHPparameters.
Remark 4:In case of short EnterHuntPhase (mode = 0), if a ticket or a card in protocol ISOB, INNOVATRON or Contact Mode has been found once it could not be seen a second time just after having been found, because the reader memorizes the serial number, as if the byte “forget” in the long EnterhuntPhase had been set to 0. If the user wants to see the card again, the forgetting of the serial number must be forced thanks to the 01_0E command.

Note : Recognition of the type of card or ticket (e.g. GTML or CD97) is the responsibility of application which then sends the commands to the card according to the appropriate class.
GEN5XX accepts GEN4XX and GEN3XX format of commands to maintain compatibility.

$01 $04

Devices: *all printer models with ASK module*

Name: End Tag Communication

Description : Ends communication with the badge or the ticket (in ISO and Innovatron protocols) and allows configuring the end of the transaction with the card.

CLASS	INS	DATA IN
$01	$04	DNX (1)

DNX : This byte is bit-mapped:
Bit0: $xxxx xxx0 does not send disconnection order, but deselects the CIDs in ISO protocol.
Bit0: $xxxx xxx1 sends disconnection order.
And for the Innovatron protocol, more options are available:
Bit1: $xxxx xx1x keeps the field ON during 60 ms after having sent, or not, the disconnection order.
Bit2: $xxxx x1xx switches OFF the field during 10ms and then switches it ON properly.
Bit3: $xxxx 1xxx forgets the last card found.
Bits7..4 are Reserved for Future Use.
In Innovatron protocol, if all the options are selected, their associated functionalities will be performed in the same order as the parametered bits, from bit 0 to bit 7.
Thus, if the user sends 01 04 0F, the coupler will:

- Send a disconnection order.
- Wait for 60 ms after having received the acknowledgement of the disconnection order.
- Switches off the field
- Switches on the field.
- Erases the Serial Number of the last card treated.

CLASS	INS	DATA OUT
$01	$04	ACK (1)

ACK : $00 disconnection acquitted
$01 disconnection not acquitted or input byte = $00

$01 $05

Devices: *all printer models with ASK module*

Name: Get Communication Status

Description : The status is that of the last exchange of radio frame; The duration of radio communication is measured between the beginnings of contact (before APGEN frame) and end of dialogue (After response to DISC or at the end of timeout of the last repetition). The number of resumptions corresponds to the difference between the number of messages sent to the card and the number of messages received from the card in the course of communication (not available for all card types).

CLASS	INS	DATA IN
$01	$05	-

No associated input data

CLASS	INS	DATA OUT			
$01	$05	STATUS (1)	TIME (2)	REP (1)	TIME_END

STATUS: communication status of last radio exchange
$01 : data received
$00 : no data received in timeout delay
$FF : data coding error
$FE : error detected by communication controller
$FD : reception buffer overflow
$FC : timeout delay expired before end of reception
$FB : CRC error

TIME : Duration in milliseconds (whole not signed, MSB at the head) of communication.

REP : Number of resumptions

TIME_END : Duration in milliseconds (whole not signed, MSB at the head) of last communication.

$01 $0E

Devices: *all printer models with ASK module*

Name: Switch off antenna
Description : switches ON or OFF the electromagnetic field of the antenna (stop carrier).

CLASS	INS	DATA IN				
$01	**$0E**	**ANT (1)**	**FIELD (2)**	**FORGET (1)**	**MODE (1)**	**TIMEOUT (1)**

ANT : RUF (selection of antenna) $00 : Antenna 1
Others : RUF
FIELD : Status of the field: $00 : OFF
$01 : ON
FORGET : $00 Memorizes the serial number of the last badge presented
Parameter kept for compatibility with previous generation couplers
$01 Forgets the serial number of the last badge presented.
MODE : Type of the present command:
$00: short (compatibility with the former one)
$01: long (takes into account the next byte for the timeout)
TIMEOUT : Duration of the present command (in ms).

(Remark : a compatibility is ensured with the old Switch_Off_Antenna format (only the first parameter for the FIELD parameter)

CLASS	INS	DATA OUT
$01	**$0E**	**STATUS (1)**

STATUS: $00 : OK
$01 : problem of communication

$01 $12

Devices: *all printer models with ASK module*

Name: Send to antenna
Description : Sends the data to the chosen transparent mode (either ISOA, ISOB, cards understanding class 5 commands, and MV5000).

CLASS	INS	DATA IN	
$01	$12	LNG (1)	DATA (x)

LNG : length of DATA
DATA : data made up of :
* The length of the command to transmit to the card + 1(actually must be equal to LNG parameter).
* The real frame to transmit to the card.

CLASS	INS	DATA OUT		
$01	$12	STATUS (1)	LNG (1)	DATA (x)

STATUS: communication status
$01 : data received
$00 (resp $03): no data received in timeout delay in Innovatron or ISOA protocol (resp. ISOB protocol)
$06 : invalid CID
$08 : ICC fails to answer correctly
$FF : data coding error
$FE : error detected by communication controller
$FD : reception buffer overflow
$FC : timeout delay expired before end of reception
$FB : CRC error
LNG : length of DATA
DATA : data made up of :
Length of the response from the card + 1.
The response from the card.

$01 $22

Devices: *all printer models with ASK module*

Name: Send to antenna extended
Description : Sends the data to the chosen transparent mode (either ISOA, ISOB, cards understanding class 5 commands, and MV5000).

CLASS	INS	DATA IN		
$01	$22	LNG LOW(1)	LNG HIGH (1)	DATA (x)

LNG LOW and LNG HIGH : length of DATA = LNG LOW + (256 * LNG HIGH)
DATA : The frame to transmit to the card.

CLASS	INS	DATA OUT			
$01	$12	STATUS (1)	LNG LOW (1)	LNG HIGH (1)	DATA (x)

STATUS: communication status
$01 : data received
$00 (resp $03): no data received in timeout delay in Innovatron or ISOA protocol (resp. ISOB protocol)
$06 : invalid CID
$08 : ICC fails to answer correctly
$FF : data coding error
$FE : error detected by communication controller
$FD : reception buffer overflow
$FC : timeout delay expired before end of reception
$FB : CRC error

LNG LOW and LNG HIGH : length of DATA = LNG LOW + (256 * LNG HIGH)
DATA : the response from the card.

$01 $13

Devices: *all printer models with ASK module*

Name: Reset SAM
Description : Executes hardware initialization of a security module.
Returns the module ATR.

CLASS	INS	DATA IN	DATA IN	DATA IN
$01	$13	SAM(1)	INN (1)	ISO (1)

SAM : Selection of SAM : $00 : SAM usually selected
$01 : SAM 1
$02 : SAM 2
$03 : SAM 3
$04 : SAM 4
Others RUF

INN : Selection of SAM in Innovatron High Speed protocol
If = $01 , selection of SAM in Innovatron protocol
If = $00, no SAM selection in this protocol

ISO : selection of protocol ISO 7816
If = $01 , selection of SAM in ISO7816 T=0 protocol
If = $02 , selection of SAM in ISO7816 T=1 protocol
If = $00 , no SAM selection in this protocol

NB1 : if these 2 parameters are positioned, there is first scrutinization in Innovatron protocol, then in ISO7816 protocol.
NB2 : If this command is 01 13 00 (INN= $01 and ISO= $00 by default)

CLASS	INS	DATA OUT		
$01	$13	STATUS (1)	LNG (1)	ATR (1)

STATUS: communication status
$00 : data received in Innovatron protocol
$80 : data received in ISO 7816 protocol.
$81 : data received in ISO 7816 protocol but writing prohibited.
$82 : data received in ISO 7816 protocol but speed is fixed at default speed.
$FF : no data received

LNG : length of data
ATR : Response to SAM RESET selected.

$01 $14

Devices: *all printer models with ASK module*

Name: Send to SAM
Description : Sends the data to the chosen transparent mode. The communication parameters are those chosen using the SAM communications parameter entry function

CLASS	INS	DATA IN			
$01	$14	SAM (1)	LNG (1)	DATA (x)	DIRECTION (1)

SAM : SAM number= $00, $01, $02, $03 or $04 as defined in "Reset Sam" command
LNG : length of data (LNG included)
DATA : data made up of the ISO command to transmit to the SAM
DIRECTION : (requisite for ISO7816 contact mode only) : IN ($01), OUT($02), IN_OUT ($03)

CLASS	INS	DATA OUT		
$01	$14	STATUS (1)	LNG (1)	DATA (x)

STATUS: communication status $00 : data received
$FF : no data received

LNG : length of data (LNG included)
DATA : data which makes up the SAM response.

$01 $15

Devices: *all printer models with ASK module*

Name: Select_CID
Description : Select a communication channel to communicate with an ISO 14443 card. parameter entry function

CLASS	INS	DATA IN
$01	$15	CID (1)

CID : CID value from 1 to MaxValue

CLASS	INS	DATA OUT
$01	$15	STATUS (1)

STATUS: Status of operation 1 = Ok, 0 = Nok (Bad CID value)

$01 $16

Devices: *all printer models with ASK module*

Name: SELECT_DIV
Description : Select a Diversifier for the transaction. Usable after a Security module change, a CID change or if the option ("no Diversifier") is chosen in the Select option command or in RS485 default option mode.

CLASS	INS	DATA IN		
$01	**$16**	**SLOT (1)**	**PROT (1)**	**CardSerialNumber (4)**

SLOT : SAM SLOT
PROT : SAM PROTOCOL (0 = Innovatron, 1 = ISO)
Card Serial Number : Last 4 bytes of the card Serial Number

CLASS	INS	DATA OUT
$01	**$16**	**STATUS (1)**

STATUS: Status of operation 1 = Ok, 0 = Nok

$01 $17

Devices: *all printer models with ASK module*

Name: EnterHuntPhase Parameters

Description : Initialize parameters for ISOA/B and Innovatron Protocol. If the first parameter is set to 00, the command can send the present configuration back.

CLASS	INS	DATA IN										
$01	$17	MAXNB CARD (1)	REQ (1)	NB SLOT(1)	AFI (1)	Auto SelDiv(1)	Deselect (1)	Select Appli(1)	LNG (1)	DATA (x)	Felica AFI(2)	Felica NBSlot(1)

MAXNBCARD : Max Number of cards to look up at (default value is 1 for single card detection)
Note: if $00 and nothing for others parameters, the present configuration will be given back.

REQ : Type Of Request (00=Req or 01=WakeUp) (default value is WakeUp)
Note : the ISO 14443 A anticollision required type of Request = Req.

NBSLOT : Number of slots in case of collision (default value is 0 for probabilistic method)

AFI : AFI value to seek (default value is 0 for All AFI)

AutoSelDiv : Automatic Select Diversifier method 1 (TRUE).

Deselect : When a later deselection will be called, if set to $01-> real deselection of the found cards and if $00-> switches the field off.

SelectAppli : Select Application mangement for ISOA / ISOB
%000x xxx1 : send Select Application to card after detection (ISOA / ISOB)
%000x xx1x : Force to $00 (instead of$94) the Select Application « CLA » field.
%000x 1xxx : add selected application name in the F01_03_EnterHuntPhase» answer.

LNG : optional data length

DATA: : optional name of the Application to select (complete or partial, in hexadecimal). Default is “1TIC.” (0x31 0x54 0x49 0x43 0x2E)

Felica AFI: Card function Identifier (defaut is all cards = 0xFFFF) Not implemented Yet

Felica Nb Slot: Slot number for Felica Anticollision (default value = 3) Not implemented Yet

CLASS	INS	DATA OUT										
$01	$17	MAXNB CARD (1)	REQ (1)	NB SLOT(1)	AFI (1)	Auto SelDiv(1)	Deselect (1)	Select Appli(1)	LNG (1)	DATA (x)	Felica AFI(2)	Felica NBSlot(1)

The Data Out represent the present configuration for the EHP.

$01 $18

Devices: *all printer models with ASK module*

Name: Switch Signals
Description : Activates or deactivates the interface signals Led

CLASS	INS	DATA IN	
$01	$18	SIGCPU	RFU

SIGCPU : 1 byte, CPU description
Bit 0 = Led 1 (red or green), activation if bit = 1, deactivation if bit = 0.
Bit 1 = Led 2 (orange or green), ditto
Bit 2 = Led 3 (green), ditto
Bit 3 = Led 4, (NE or green), ditto

NB: this byte inactive if the AUTO LEDs configuration = 1

RFU : Kept for compatibility this parameter is not taken into account in GEN5XX
(N.B.: There is no integrated MUX 482 management. Anyway it is still possible to manage this peripheral in other ways on demand).

CLASS	INS	DATA OUT
$01	$18	STATUS (1)

STATUS: status = $00

$01 $19

Devices: *all printer models with ASK module*

Name: Select SAM
Description : Selects the SAM chosen for the application

CLASS	INS	DATA IN	
$01	$19	SAM (1)	Protocol (1)

SAM : SAM slot number from 1 to 4
Protocol : 0 : Innovatron protocol
1 : ISO7816 T=0 protocol
2: ISO7816 T=1 protocol

CLASS	INS	DATA OUT
$01	$19	STATUS (1)

STATUS: communication status :
$00 : command possible
$other : SAM absent

$01 $23 (not implemented yet)

Devices: *all printer models with ASK module*

Name: Buzzer management
Description : Sounds the buzzer

CLASS	INS	DATA IN		
$01	**$23**	**TYPE (1)**	**Frequency (2)**	**Duration (2)**

TYPE : 0 = system beep (500 Hz, 250 ms)
1 = PayPass sequence (Buzzer + 4 leds)
2 = user beep (see parameters below)
Frequency : MSB LSB = frequency in Hz
Duration : MSB LSB = duration in ms

CLASS	INS	DATA OUT
$01	**$23**	**STATUS (1)**

STATUS: communication status : always 00

$01 $24

Devices: *all printer models with ASK module*

Name: Extended IO Configuration
Description : Configure extended IO in input or output
Each input IO field is a bitmap definition of the IO as follow:

Bit_7	Bit_6	Bit_5	Bit_4	Bit_3	Bit_2	Bit_1	Bit_0
IO1_UCM108	IO2_UCM108	IO3_UCM108	IO4_UCM108	IO5_UCM108	0	0	0

CLASS	INS	DATA IN						
$01	**$24**	**IO input mask**	**Enable Pullup**	**Enable Filter**	**IO Output mask**	**IO output default value**	**IO output Enable OpenDrain**	**IO output Enable Pullup**

IO input mask : IO Bitmap to configure in input
IO input Enable PullUp mask : Enable PullUp for masked given input I/O
IO input Enable Filter mask : Enable Filter for masked given input I/O
IO output mask : IO Bitmap to configure in output
IO output default value : Default output masked value
IO output Enable OpenDrain : Enable OpenDrain for masked given output I/O
IO output Enable PullUp : Enable PullUp for masked given output I/O

N.B. : IO input Enable Filter mask override the value of IO input Enable PullUp mask
N.B. : IO output Enable OpenDrain override the value of IO output Enable PullUp mask

Response:

CLASS	INS	DATA OUT
$01	**$24**	**STATUS (1)**

$01 $25

Devices: *all printer models with ASK module*

Name: Extended IO Read
Description : Read the Extended IO word
Each input IO field is a bitmap definition of the IO as follow:

Bit_7	Bit_6	Bit_5	Bit_4	Bit_3	Bit_2	Bit_1	Bit_0
IO1_UCM108	IO2_UCM108	IO3_UCM108	IO4_UCM108	IO5_UCM108	0	0	0

CLASS	INS	DATA IN
$01	**$25**	**Mask on IO bitmap word**

Mask on IO bitmap : Word on the IO concerned

N.B. : Bit 0 to 2 are reserved for future use (input value are not taken into account and output values are non significant).

Response :

CLASS	INS	DATA OUT
$01	**$25**	**Bitmap on IO word read**

BITMAP of IO word read : IO read masked by input parameter.

$01 $26

Devices: *all printer models with ASK module*

Name: Extended IO Write
Description : Write on the Extended IO word
Each input IO field is a bitmap definition of the IO as follow:

Bit_7	Bit_6	Bit_5	Bit_4	Bit_3	Bit_2	Bit_1	Bit_0
IO1_UCM108	IO2_UCM108	IO3_UCM108	IO4_UCM108	IO5_UCM108	0	0	0

CLASS	INS	DATA IN	
$01	**$26**	**Mask on IO bitmap word**	**Mask on Desired value**

Mask on IO bitmap : Gives a word on the IO concerned
Mask on IO desired value : values desired on IO ports according to the mask on IO bitmap

N.B. : Bit 0 to 2 are reserved for future use (input value are not taken into account).

Response :

CLASS	INS
$01	**$26**

$01 $27

Devices: *all printer models with ASK module*

Name: Set SAM Baudrate PPS
Description : Perform a PPS

CLASS	INS	DATA IN	
$01	**$27**	**ProProt**	**ParamFD**

ProProt : Proposed protocol (0 for T=0; 1 for T=1)
ParamFD : FiDi parameter

Response :

CLASS	INS	DATA OUT	
$01	**$27**	**Status**	**Status**

STATUS: 2 bytes status

3.7.3 GTML Class (N°= $02)

Presentation of GTML badge (Generic Transport Mask Light):
This Card is a transport-dedicated card.

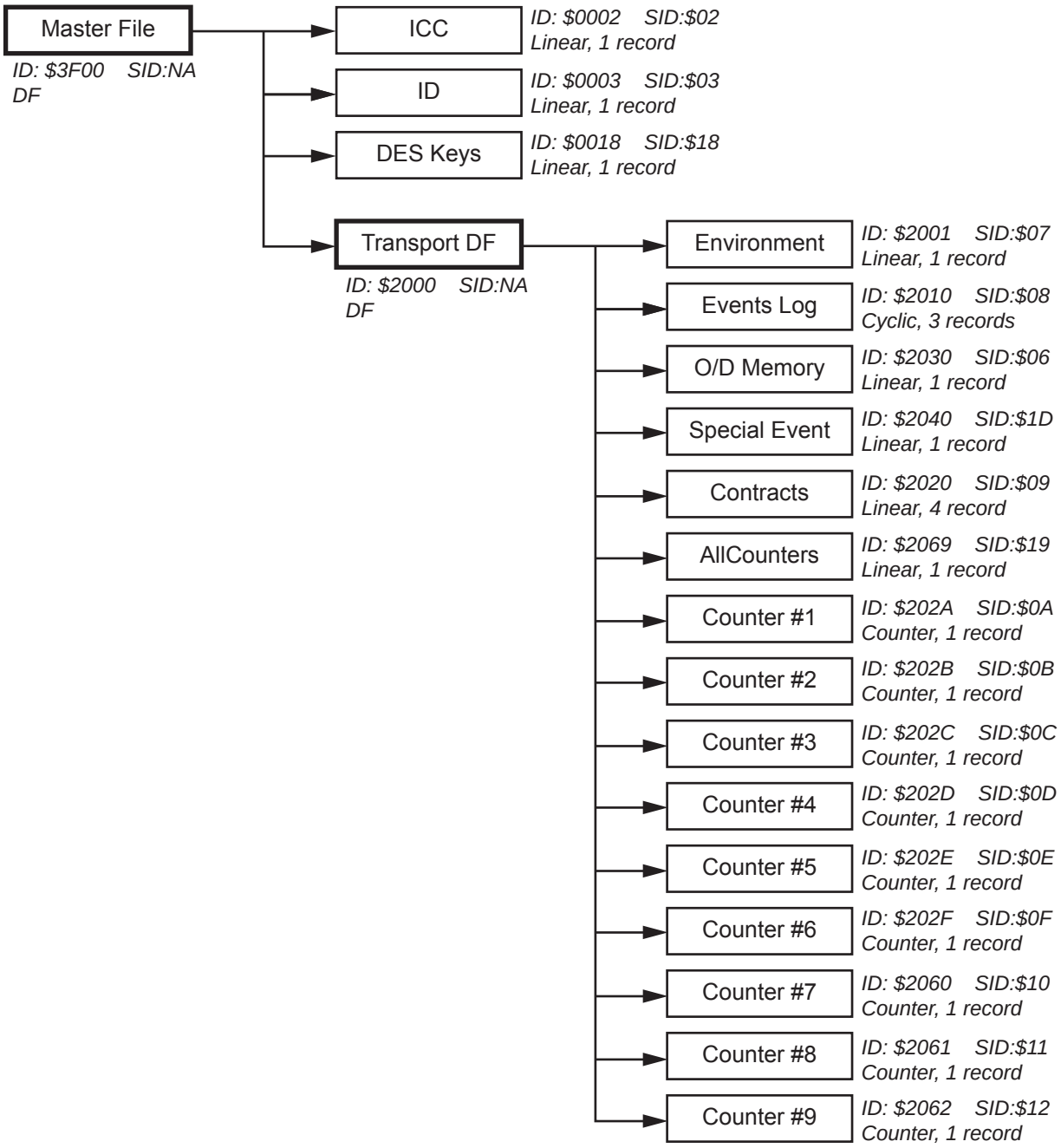
3.7.3.1 Main functions

Ticketing : ensures checking of access to public transport networks. This is its main function. This includes the different existing pricing modes, and should enable the introduction of new ones.

3.7.3.2 Structure of data

The card is organized into files according to ISO/IEC 7816-4 and takes its inspiration from the EN726-3 norm. The GTML card files have several attributes :

- type : DF (dedicated file) or EF (elementary file),
- long identifier (LID),
- short identifier(SID),
- sub type (only for the EFs) : fixed structure, circular, counter.



3.7.3.3 Functionalities

Definitions common to class functions :

Access mode:

Value on 1 byte defining the authentication principles used to access data.

DEFAULT	$00	No local cryptogram for the function. If a session is open, it is the session security that is used. Otherwise only data of which the attribute is "ALWAYS" will be accessible.

Concerning GTML the only authentication mode possible for recording is the session mode, and for reading it is “always” mode or the session mode.

Execution report :

Value on 3 bytes (1 general byte, 2 bytes of detail).

Title	byte 0	Bytes 1-2 (detail)	
Execution carried out (no error)	$00	$90 $00	Command accepted and executed.
File error	$01	$69 $81	Incompatibility with file structure.
		$6A $82	File not found
		$6A $83	Recording not found.
Badge side security error	$02	$69 $82	Security conditions not respected (no alea, cryptogram absent, no current session, access condition unknown)
		$63 $CX	Incorrect signature. X represents the number of attempts still authorized (0, 1 or 2)
		$69 $83	Command rejected since maximum number of errors (3) was reached (PIN access only).
Session Error	$03	$64 $00	EEPROM capacity insufficient for the session.
		$69 $85	Command forbidden(the mode NEVER is indicated for this command).
		$69 $86	Command prohibited on a DF.
Badge physical Error	$04	$62 $81	The data item returned is corrupted (Fatal Error)
		$65 $81	EEPROM Failure (Fatal Error).
Application Error	$05	$6A $80	Value to deduct or add incorrect.
		$6A $81	Wrong key level specified (1 to 3).
		$6A $87	Lc incompatible with P1-P2
		$67 $00	P3 invalid (returned if P3 = 0).
		$6B $00	P1 – P2 not supported.
		$6D $00	INS not supported.
Invalid card	$06	$6E $00	CLA not supported.
Badge error code unknown	$09	Badge return code unknown	
Abnormal execution reported by security module	codes $20-$29	cf spec CD97MiniMS	
SAM side security alert	$20	$69 $00	Command not authorized (A counter of key uses reached its maximum value)
		$69 $85	Command not authorized (conditions of use nor satisfied)
		$69 $88	Cryptogram incorrect
SAM physical error	$21	$65 $81	Eeprom Problem
SAM instruction error	$22	$94 $10	Value incorrect in incoming data.
		$94 $20	Exceeding of PME badge capacity.
		$6A $83	Recording not found. The key requested is not present in the key file .
		$6A $86	P1-P2 incorrect. Key reference too large.
		$6A $87	Lc incompatibly with P1-P2.
		$64 $00	Execution error.
SAM security alert	$29	SAM return code unknown	
SAM Answer TimeOut	$2F	$00 $EF	No SAM answer
Incorrect card communication (dialogue lost)	$40	byte 0 :	$00 (RUF)
		byte 1 :	INS code (ISO) of the frame which has failed ($00) if multiple command.
Incorrect SAM communication	$41	byte 0 :	$00 (RUF)
		byte 1 :	INS code (ISO) of the frame which has failed.

Abnormal execution reported by the CSC controller	$80	$00$00	command unknown$00$01 concatenation not authorized for this function
		$00$02	maximum number of concatenable functions reached
		$00$03	access mode incompatible with the concatenation
		$00$04	access mode prohibited for this function
		$00$05	no session open
		$00$06	session already open
		$00$07	badge response incorrect
		$00$08	command prohibited in session
		$00$09	overflow of transmission buffer to the badge
		$00$0A	SAM key reference version "x" not found
		$00$0B	ChangeKey prohibited if not preceded by SelectFile
		$00$0C	ChangeKey prohibited when the new version of key is 0
		$00$0D	the attempt to ChangeKey with a key version >1 failed since the key in the card is version 0 (personalization to do)
		80xx	error in index input parameter $xx

Note : These report values are proper to the class.

Structure of elementary counter files :

Value : 3 bytes, binary unsigned, representing current counter value.
Free data : 5 bytes, Value fixed at $0000000000.
Floor : 3 bytes, representing counter floor (its value is fixed at $000000).
Ceiling : 3 bytes, representing counter ceiling(its value is fixed at $FFFFFF).

FCI (file description data) :

Tag 1 byte, Fixed value = $85
Length 1 byte, Fixed value = $17
SID 1 byte, short identifier, = $00 for the DF and for the MF.
Type 1 byte : = $01 for MF
= $02 for DF
= $04 for EF
EFType 1 byte, type of elementary file = $00 for the DF and for the MF.
= $02 Linear fixed structure.
= $04 Circular.
= $08 Counter.
RecSize 1 byte, number of bytes per recording.
= $00 for the DF and MF.
= $1D for the EF.
NumRec 1 byte, number of recordings in the file,
= $00 for the DF and MF.
AC 4 bytes : the 1st for the command index number 0, the 2nd for the command index number 1, the 3rd for the command index number 2 and 4th for the command index number 3.

The possible access modes are:
= $00 NEVER
= $01 PIN
= $10 SESSION
= $1F ALWAYS
Other values RUF.

Correspondence chart between command index number and type of file.

Command index number	MF and DF	EF
0	REHABILITATE	Read Record
1	INVALIDATE	Update Record
2	RUF	Write Record or Decrease
3	RUF	Append Record or Increase

Nkey 4 bytes, key index number to be used.
Status 1 byte :
$X0 Card Valid.
$X1 Card invalidated.
$0X No errors on PIN presentation.
$1X An error of PIN presentation.
$3X Two errors of PIN presentation.
$7X Three errors of PIN presentation. Any new presentation of PIN is rejected.
Other values RUF.
KVC1 1 byte, =$00 for an EF.
KVC2 1 byte, =$00 for an EF.
KVC3 1 byte, =$00 for an EF.
Floor 3 bytes, Value fixed at $000000.
Ceiling 3 bytes, Value fixed at $FFFFFF for the counters and at $000000 for the other types of file.

3.7.3.4 Set of instruction

$02 $01

Devices: *all printer models with ASK module*

Name: APPEND RECORD (GTML)
Description : Adds a record to an elementary circular file.
This new record becomes the first in the file, the older one is suppressed.

CLASS	INS	DATA IN			
$02	$01	ACCES = $00	SID (1)	LNG (1)	DATA (x)

ACCES : access mode = $00 (other RFU values).
SID : SID (SID = $00 for usually selected EF)
LNG : length of data to record (n bytes £ size of a recording)
DATA : data to record

CLASS	INS	DATA OUT
$02	$01	REND (3)

REND : execution report

$02 $06

Devices: *all printer models with ASK module*

Name: READ RECORD (GTML)
Description : Reading of a record given in a circular EF, a counter or an EF linear fixed structure.

CLASS	INS	DATA IN			
$02	$06	ACCES = $00	SID (1)	NREC (1)	LNG (x)

ACCES : access mode =$00 (other RFU values).
SID : SID (SID = $00 for usually selected EF)
NREC : recording number
LNG : length of data to read

CLASS	INS	DATA OUT	
$02	$06	REND (3)	DATA (x)

REND : execution report
DATA : data read (n bytes)

$02 $02

Devices: *all printer models with ASK module*

Name: CHANGE PIN (GTML)
Description : Records PIN value contained in the MF.
A SelectFile of the MF should be carried out before being able to change the PIN.
This command can't be used during a session.

CLASS	INS	DATA IN			
$02	$02	$04	RFU (1)	OLDPIN (4)	NEWPIN (4)

RFU : For future use
OLDPIN : old PIN
NEWPIN : new PIN

CLASS	INS	DATA OUT
$02	$02	REND (3)

REND : execution report

$02 $0B

Devices: *all printer models with ASK module*

Name: VERIFY PIN (GTML)
Description : Presentation of PIN with counting of number of incorrect presentations.
A SelectFile of the MF should be carried out before being able to check PIN.

CLASS	INS	DATA IN	
$02	$0B	MODE (1)	PIN (4)

MODE : $00, consultation of counter of incorrect presentations
$01, presentation of PIN
$02, presentation of PIN in transparent mode for contact communication
PIN : PIN (4 bytes)

CLASS	INS	DATA OUT
$02	$0B	REND (3)

REND : execution report

$02 $03

Devices: *all printer models with ASK module*

Name: DECREASE (GTML)
Description : Decreases the value contained in a counter file.
Remark : This function should be carried out in session and it immediately returns the new value of the counter.

CLASS	INS	DATA IN			
$02	$03	ACCES = $00	SID (1)	VALUE (3)	$00 $00 $00 $00 $00

ACCES : access mode = $00 (other RFU values).
SID : SID (SID = $00 for usually selected EF)
VALUE: value to deduct (3 bytes, binary number positive or nil)

CLASS	INS	DATA OUT	
$02	$03	REND (3)	VALUE (3)

REND : execution report
VALUE: new value (3 bytes, binary number signed)

$02 $04

Devices: *all printer models with ASK module*

Name: INCREASE (GTML)
Description : Increases the value contained in a counter file.
The associated data is not written in the GTML.
Remark : This function should be carried out in session and it immediately returns the new value of the counter.

CLASS	INS	DATA IN			
$02	$04	ACCES = $00	SID (1)	VALUE (3)	$00 $00 $00 $00 $00

ACCES : access mode = $00 (other RFU values).
SID : SID (SID = $00 for usually selected EF)
VALUE: value to add (3 bytes, binary number positive or nil)

CLASS	INS	DATA OUT	
$02	$04	REND (3)	VALUE (3)

REND : execution report
VALUE: new value (3 bytes, binary number signed)

$02 $05

Devices: *all printer models with ASK module*

Name: INVALIDATE (GTML)
Description : Invalidates current DF(and consequently all the related files).

CLASS	INS	DATA IN
$02	$05	ACCES = $00

ACCES : access mode = $00 (other RFU values).

CLASS	INS	DATA OUT
$02	$05	REND (3)

REND : execution report

$02 $07

Devices: *all printer models with ASK module*

Name: REHABILITATE (GTML)
Description : Cancels file invalidation.

CLASS	INS	DATA IN
$02	$07	ACCES = $00

ACCES : access mode = $00 (other RFU values).

CLASS	INS	DATA OUT
$02	$07	REND (3)

REND : execution report

$02 $08

Devices: *all printer models with ASK module*

Name: SELECT FILE (GTML)
Description : Explicit selection of current EF or DF.
This command sends back file description data.

CLASS	INS	DATA IN		
$02	$08	CNTR (1)	LNG (1)	PATH (x)

CNTR : Control of selection
$00, MF
$02, EF in current DF (identifier bytes 2 to n)
$08, bytes 1 to n : path from the MF (excluded)
LNG : length of identifier or path
PATH : identifier or path

NB: The command $02 $08 $00 $00 is not supported by CD97 card.

CLASS	INS	DATA OUT	
$02	$08	REND (3)	FCI (x)

REND : execution report
FCI : FCI

$02 $0A

Devices: *all printer models with ASK module*

Name: UPDATE RECORD (GTML)
Description : Deletion then writing of a record given in a linear fixed structure EF or the most recent recording of a circular file.

CLASS	INS	DATA IN				
$02	$0A	ACCES = $00	SID(1)	NREC (1)	LNG (1)	DATA (x)

ACCES : access mode =$00 (other RFU values).
SID : SID (SID = $00 for usually selected EF)
NREC : recording number
LNG : number of bytes to write (n bytes £ size of a recording)
DATA : data to record

CLASS	INS	DATA OUT
$02	$0A	REND (3)

REND : execution report

$02 $0C

Devices: *all printer models with ASK module*

Name: WRITE RECORD (GTML)
Description : Writing without deletion of a record given in a linear fixed structure EF or the most recent recording of a circular file.

CLASS	INS	DATA IN				
$02	$0C	ACCES = $00	SID (1)	NREC (1)	LNG (1)	DATA (x)

ACCES : access mode =$00 (other RFU values).
SID : SID (SID = $00 for usually selected EF)
NREC : recording number ($01 for a circular file)
LNG : number of bytes to write (n bytes £ size of a recording)
DATA : data to record

CLASS	INS	DATA OUT
$02	$0C	REND (3)

REND : execution report

$02 $10

Devices: *all printer models with ASK module*

Name: OPEN SECURED SESSION (GTML)
Description : Opening of a certification session.
Returns the paths of the DF (from the MF excluded) of non-ratified applications and the data read in the indicated recording and file.

CLASS	INS	DATA IN		
$02	$10	TYPE (1)	SID (1)	NREC (1)

TYPE : Type of operation
$00 : personalization
$01 : reloading
$02 : validation
SID : SID (SID = $00 for usually selected EF)
NREC : recording number to read
$00 : no recording to read
$01 : reading of a number 1 or unique recording

CLASS	INS	DATA OUT					
$02	$10	REND (3)	NBAPP (1)	PATH1 (2)	...	PATHn (2)	DATA (29)

REND : execution report
NBAPP: 2*n where n is the number of non-ratified applications
PATH1 : path of the first non-ratified application
....
PATHn: path of the n^{th} non-ratified application
DATA : if recording to read, 29 bytes of data

$02 $11

Devices: *all printer models with ASK module*

Name: CLOSE SECURED SESSION (GTML)
Description : Closure of the certification session

CLASS	INS	DATA IN
$02	$11	-

No associated input data

CLASS	INS	DATA OUT
$02	$11	REND (3)

REND : execution report

$02 $12

Devices: *all printer models with ASK module*

Name: ABORT SECURED SESSION (GTML)
Description : Stop the current certification session. This still allow to continue to dialogue with the badge and, in particular, open a new session.
Note : this function is emulated in the case of the GTML.

CLASS	INS	DATA IN
$02	$12	-

No associated input data

CLASS	INS	DATA OUT
$02	$12	REND (3)

REND : execution report

3.7.4 CD 97 Class (N°= $03)

Presentation of CD97 Card :
The Travel Card 1997 is a remote multi-application ticket card destined for payment of transport and other services.

3.7.4.1 Main functions

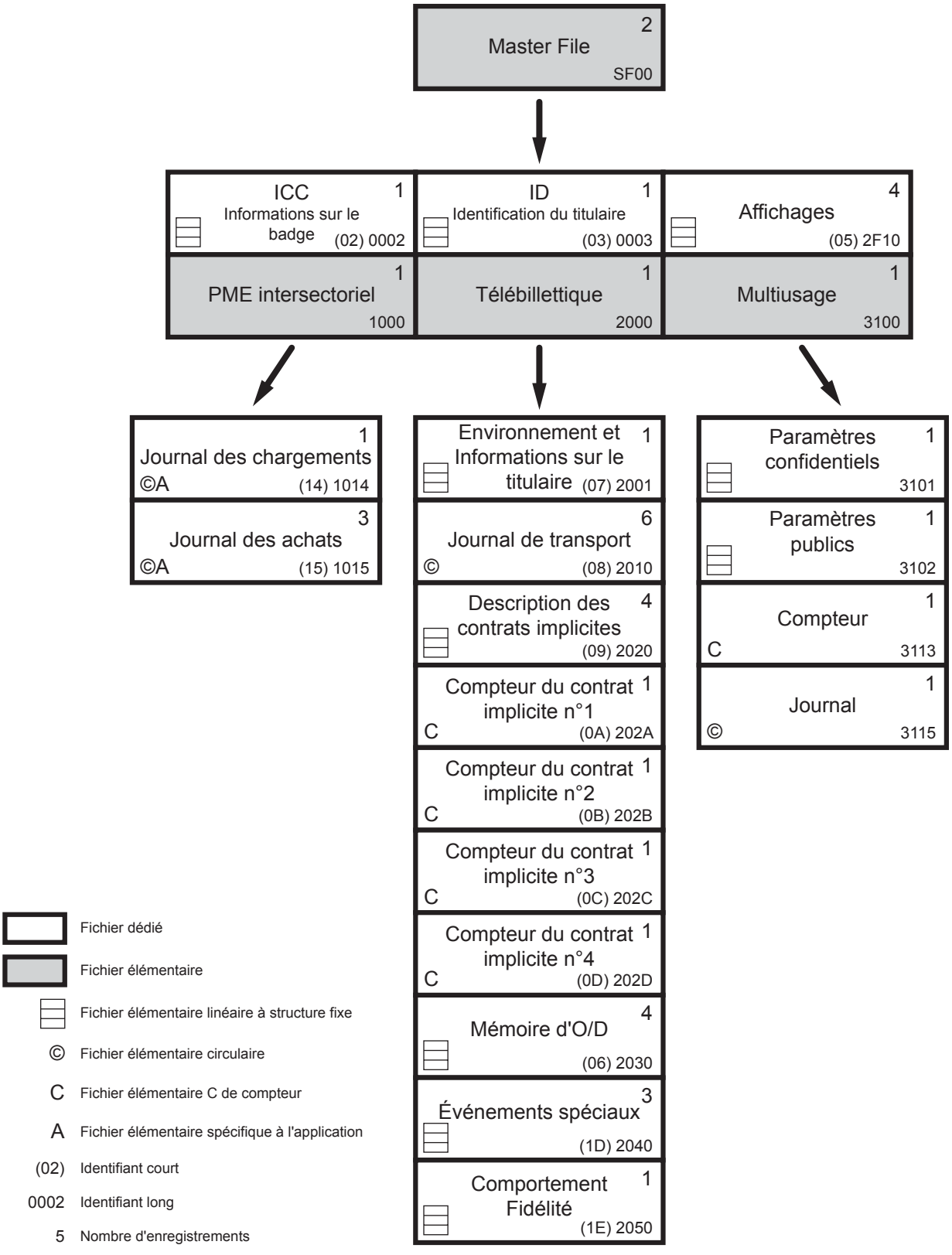
1. Ticketing : handle control of access to public transport networks. This is its main function. This includes the different existing modes of pricing, and should enable the introduction of new ones.
2. Payment : could be used as a means of payment for operators not linked to transport (France Télécom, Ville de Paris(Paris City Hall), Relais H(chain of newspaper stands), etc.).
3. Private : could be used as private badge by any operator, for example, for control of access.

3.7.4.2. Data structure

The card is organized into files according to the ISO/IEC 7816-4 and takes its inspiration from the EN726-3 norm.
The CD97 card files have various attributes :

- type : DF (dedicated file) or EF (elementary file),
- long identifier (LID),
- short identifier (SID),
- sub type (solely for the EFs) : fixed structure, circular, counter.

The DFs are proper to an application (payment, transport, etc...), they are like folders which contain the EFs.


2
Master File
SF00
ICC 1
Informations sur le badge
(02) 0002
ID 1
Identification du titulaire
(03) 0003
4
Affichages
(05) 2F10
1
PME intersectoriel
1000
1
Télébillettique
2000
1
Multiusage
3100
1
Journal des chargements
©A
(14) 1014
3
Journal des achats
©A
(15) 1015
Environnement et Informations sur le titulaire 1
(07) 2001
6
Journal de transport
©
(08) 2010
Description des contrats implicites 4
(09) 2020
Compteur du contrat implicite n°1 1
C
(0A) 202A
Compteur du contrat implicite n°2 1
C
(0B) 202B
Compteur du contrat implicite n°3 1
C
(0C) 202C
Compteur du contrat implicite n°4 1
C
(0D) 202D
Mémoire d'O/D 4
(06) 2030
Événements spéciaux 3
(1D) 2040
Comportement Fidélité 1
(1E) 2050
Paramètres confidentiels 1
3101
Paramètres publics 1
3102
Compteur 1
C
3113
Journal 1
©
3115
Fichier dédié
Fichier élémentaire
Fichier élémentaire linéaire à structure fixe
© Fichier élémentaire circulaire
C Fichier élémentaire C de compteur
A Fichier élémentaire spécifique à l'application
(02) Identifiant court
0002 Identifiant long
5 Nombre d'enregistrements

Definitions common to class functions :

Access mode :

Value on 1 byte defining the authentication principles used to access data.

DEFAULT	$00	No local cryptogram for the function. If a session is open, it is the session security that is used. Otherwise only data of which the attribute is "ALWAYS" will be accessible.
PROTECTED	$01	Ground authentication, of incoming data for the badge. Immediate execution by the badge.
STAMPED	$02	Authentication of the badge, of outgoing data for the badge.

Remark : it is impossible to combine the mode « protected » or the mode « stamped » with the session mode.

Execution report :

Value on 3 bytes (1 general byte, 2 bytes of details).

Title	byte 0	Bytes 1-2 (detail)	
Execution carried out (no error)	$00	$62 $00	Command accepted execution conditional on good session outcome.
		$90 $00	Command accepted and executed.
Abnormal execution reported by badge	codes $01-$09	cf. spec. CD97	
File error	$01	$62 $82	End of recording attained.
		$69 $81	Incompatibility with file structure.
		$6A $82	File not found
		$6A $83	Recording not found.
		$6A $84	Memory insufficient in the file.
		$98 $10	Application DF or MF invalidated.
Badge side security error	$02	$69 $82	Security conditions non respected (no alea, cryptogram absent, no current session, access condition unknown)
		$63 $00	Certificate incorrect in protected mode.
		$63 $CX	Signature incorrect. X represents the number of attempts still authorized (0, 1 or 2)
		$69 $83	Command rejected since maximum number of errors (3) was reached (PIN access only)
		$98 $00	PME Certificate incorrect.
Session Error	$03	$69 $86	Command not authorized (no current EF).
		$64 $00	EEPROM capacity insufficient for the session.
		$69 $85	Access prohibited with this command.
		(mode NEVER)	Session Problem.
Badge physical error	$04	$65 $81	Memory failure
		$96 $10/20/30	EEPROM Failure.

Error application	$05	$6A $80	Value to deduct or add incorrect. PIN incorrect
		$6A $81	function not handled.
		$6A $87	Lc incompatible with P1-P2
		$6B$00	P1 or P2 incorrect
		$98 $30	P2 incorrect
		$98 $40	P2 of previous GetEP incorrect (or GetEP omitted)
		$98 $60	Operation impossible, PME already at minimum or PME capacity exceeded.
		$67 $00	P3 incorrect
		$68 $00	Unsupported on CD97 and GTML (KVC request on Open Secured Session)
		$6D $00	INS incorrect
Card invalid	$06	$62 $83	DF parent or MF invalidated.
Badge error code unknown	$09	Badge return unknown code	
Abnormal execution reported by security module	codes $20-$29	cf. spec CD97MiniMS	
SAM side security alert	$20	$69 $00	Command not authorized (A counter of key uses or the n° of PME transactions attained its maximum
		$69 $85	Command not authorized (conditions of use nor satisfied)
		$69 $88	Cryptogram incorrect
SAM physical error	$21	$65 $81	Eeprom Problem
SAM instruction error	$22	$94 $10	Value incorrect in incoming data.
		$94 $20	Exceeding of PME badge capacity.
		$6A $83	Recording not found. The key requested is not present in the key file .
		$6A $86	P1-P2 incorrect. Key reference too large.
		$6A $87	Lc incompatibly with P1-P2.
		$64 $00	Execution error.
SAM security alert	$29	SAM return unknown code	
SAM Answer TimeOut	$2F	$00 $EF	No SAM answer
Communication incorrecte badge (abandon dialogue)	$40	byte 0 :	$00 (RUF)
		byte 1 :	INS code (ISO) of the frame which has failed ($00) if multiple command.
SAM communication incorrect	$41	byte 0 :	$00 (RUF)
		byte 1 :	INS code (ISO) of the frame which has failed.
Abnormal execution reported by the CSC controller	$80	$00$00	command unknown$00$01 concatenation not authorized for this function
		$00$02	maximum number of concatenable functions attained
		$00$03	access mode incompatible with the concatenation
		$00$04	access mode prohibited for this function
		$00$05	no session open
		$00$06	session already open
		$00$07	badge response incorrect
		$00$08	command prohibited in session
		$00$09	overflow of transmission buffer to the badge
		$00$0A	SAM key reference version "x" not found
		$00$0B	Change Key prohibited if not preceded by SelectFile
		$00$0C	Change Key prohibited when the new version of key is 0
		$00$0D	the attempt to ChangeKey with a key version >1 failed since the key in the card is version 0 (personalization to do)
		80xx	error in index input parameter $xx

Note : These report values are proper to the class.

Structure of elementary counter files :

Value : 3 bytes, binary unsigned, representing current counter value.
Free data : 5 bytes, Value fixed at $0000000000.
Floor : 3 bytes, representing counter floor (its value is fixed at $000000).
Ceiling : 3 bytes, representing counter ceiling(its value is fixed at $FFFFFF).

FCI (file description data) :

Tag $85
Length $17

Value :

SID 1 byte, short identifier, =$00 for the DF and for the MF.
Type 1 byte :
$01 MF
$02 DF
$04 EF
EFType 1 byte, type of elementary file, =$00 for the DF and for the MF.
$02 Linear fixed structure.
$04 Circular.
$08 Counter.
$10 Specific to the application.
RecSize 1 byte, number of bytes per recording.
=$00 for the DF and MF.
DataSize 1 byte, number of recordings, =$00 for the DF and MF.
AC 4 bytes : the 1st for the command index number 0, the 2nd for the command index number 1, the 3rd for the command index number 2 and 4th for the command index number 3.

The possible access modes are:
$00 NEVER
$01 PIN
$02 PROTECTED
$0F APPLICATIVE
$12 PROTECTED ^ SESSION
$1F ALWAYS
Other RUF values.

Correspondence chart between command index number and type of file.

Command index number	MF and DF	Linear fixed structure EF	EF circular	Counter EF	circular EF specific to the application
0	REHABILITATE	READ RECORD	READ RECORD	READ RECORD	GetEP and READ RECORD
1	INVALIDATE	UPDATE RECORD	UPDATE RECORD	UPDATE RECORD	ReloadEP
2	-	WRITE RECORD	-	DECREASE	DebitEP and UnDebitEP
3	-	-	APPEND RECORD	INCREASE	-

Nkey	4 bytes, key index number to use.
Status	1 byte : $X0 File valid(still readable if invalidated). $X1 File invalidated, readable. $X2 File valid(not readable if invalidated). $X3 File invalidated, not readable. $0X No errors of PIN presentation. $1X One error of PIN presentation. $3X Two errors of PIN presentation. $7X Three errors of PIN presentation. Any new presentation of the PIN is rejected. Other RUF values.
KVC1	1 byte, =$00 for an EF.
KVC2	1 byte, =$00 for an EF.
KVC3	1 byte, =$00 for an EF.
Floor	3 bytes, =$000000 for files other than counter EFs.
Ceiling	3 bytes, =$000000 for files other than counter EFs.

3.7.4.3 Set of instructions

$03 $01

Devices: *all printer models with ASK module*

Name: APPEND RECORD (CD97)
Description : Adds a record to an elementary circular file.
This new record becomes the first in the file, the older one is suppressed.

CLASS	INS	DATA IN			
$03	$01	ACCES (1)	SID (1)	LNG (1)	DATA (x)

ACCES : access mode
SID : SID (SID = $00 for usually selected EF)
LNG : length of data to record (n bytes £ size of a recording)
DATA : data to record

CLASS	INS	DATA OUT
$03	$01	REND (3)

REND : execution report

$03 $02

Devices: *all printer models with ASK module*

Name: CHANGE KEY (CD97)
Description : Writes the values of a key in current DF, contained in the MF.
A SelectFile of the MF or DF should be carried out before being able to change a key.
This command can't be used during a session.

Data In:

CLASS	INS	DATA IN						
$03	$02	IKEY (1)	NKEY (1)	TYPE_CMD (1)	CKEY (1)	ALG_TAG (1)	ALG_SAM (1)	IKEY_TAG (1)

IKEY : index number of the key to modify ($01-$03) (for CD97, GTML)
Or index number (in the SAM) of the new key to be loaded in the card
NKEY : new version of the key to modify (different from 0)
For the Personalization, the new key version is $01
TYPE_CMD : $00, short command (compatibility with the former one)
$01, long command
CKEY : Index Number (in the SAM) of the key to encipher the transfer
ALG_TAG: algorithm key card to recopy
ALG_SAM: algorithm of the SAM used
IKEY_TAG: index number of the new key in the card in the DF

Data Out:

CLASS	INS	DATA OUT
$03	$01	REND (3)

REND : execution report

$03 $02 $04

Devices: *all printer models with ASK module*

Name: CHANGE KEY (CD97)

Description : Writes the values of a key in current DF, contained in the MF.
A SelectFile of the MF or DF should be carried out before being able to change a key.
This command can't be used during a session.

Data In:

CLASS	INS	DATA IN								
$03	$02	IKEY $04	KEY_NUM (1)	OLDPIN (4)	NEWPIN (4)	TYPE_CMD (1)	NKEY/KIF (1)	00/ KVC (1)	ALG (1)	NSAM (1)

IKEY : $04

KEY_NUM: key number ($00 for CD97, GTML and CT2000,
$04 for GTML2 and CD21,
$09 for POPEYE)

OLDPIN : old PIN (used only for CD97)

NEWPIN : new PIN

TYPE_CMD : $00, short command (compatibility with the former one)
$01, long command

NKEY/ KIF: SAM key number to use
or KIF of the key.

00/ KVC: $00 (if NKEY passed in the previous parameter)
or KVC of the Key.

ALG: algorithm of the SAM used (not used for CD97, GTML1 and GTML2).

NSAM : SAM number
$00 : default SAM,
$01, $02, $03 or $04 : logical number of the wanted SAM number

Data Out:

CLASS	INS	DATA OUT
$03	$02	REND (3)

REND : execution report

$03 $03

Devices: *all printer models with ASK module*

Name: APPEND RECORD (CD97)
Description : Adds a record to an elementary circular file.
This new record becomes the first in the file, the older one is suppressed.

CLASS	INS	DATA IN			
$03	$03	ACCES (1)	SID (1)	VALUE (3)	$00 $00 $00 $00 $00

ACCES : access mode
SID : SID (SID = $00 for usually selected EF)
VALUE: value to deduct (3 bytes, binary number positive or nil)

CLASS	INS	DATA OUT	
$03	$01	REND (3)	VALUE (3)

REND : execution report
outside session:
VALUE: new value (3 bytes, binary number signed)

$03 $04

Devices: *all printer models with ASK module*

Name: INCREASE (CD97)
Description : Increases the value contained in a counter file.
Records the associated data.
Remark : Executed in session, this function does not return the new value of the counter ; it will be returned by the CloseSecuredSession function

CLASS	INS	DATA IN			
$03	$04	ACCES (1)	SID (1)	VALUE (3)	$00 $00 $00 $00 $00

ACCES : access mode
SID : SID (SID = $00 for usually selected EF)
VALUE: value to add (3 bytes, binary number positive or nil)

CLASS	INS	DATA OUT	
$03	$04	REND (3)	VALUE (3)

REND : execution report
outside session:
VALUE: new value (3 bytes, binary number signed)

$03 $05

Devices: *all printer models with ASK module*

Name: INVALIDATE (CD97)
Description : Invalidates the current DF (and consequently all related files).

CLASS	INS	DATA IN
$03	$05	ACCES (1)

ACCES : access mode

CLASS	INS	DATA OUT
$03	$05	REND (3)

REND : execution report

$03 $07

Devices: *all printer models with ASK module*

Name: REHABILITATE (CD97)
Description : Cancels file invalidation.

CLASS	INS	DATA IN
$03	$07	ACCES (1)

ACCES : access mode

CLASS	INS	DATA OUT
$03	$07	REND (3)

REND : execution report

$03 $06

Devices: *all printer models with ASK module*

Name: READ RECORD (CD97)
Description : Reading of a record given in a circular EF, a counter or a linear fixed structure EF

CLASS	INS	DATA IN			
$03	$06	ACCES (1)	SID (1)	NREC (1)	LNG (1)

ACCES : access mode
SID : SID (SID = $00 for usually selected EF)
NREC : recording number
LNG : length of data to read

CLASS	INS	DATA OUT	
$03	$06	REND (3)	DATA (x)

REND : execution report
DATA : data read (n bytes)

$03 $08

Devices: *all printer models with ASK module*

Name: SELECT FILE (CD97)
Description : Explicit selection of current EF or DF.
This command sends back the file description data.

CLASS	INS	DATA IN		
$03	$06	CNTR (1)	LNG (1)	PATH (x)

CNTR : Selection check
$00, MF
$02, EF in the current DF (identifier bytes 2 to n)
$08, bytes 1 to n : path from MF (excluded)
LNG : length of identifier or of path
PATH : identifier or path

NB: The command $03 $08 $00 $00 is not supported by CD97 card.

CLASS	INS	DATA OUT	
$03	$08	REND (3)	FCI(x)

REND : execution report
FCI : FCI

$03 $09

Devices: *all printer models with ASK module*

Name: STATUS (CD97)
Description : ditto SELECT FILE, but without selecting a file.

CLASS	INS	DATA IN		
$03	$09	CNTR (1)	LNG (1)	PATH (x)

CNTR : Selection check
$00, MF
$02, EF in the current DF (identifier bytes 2 to n)
$08, bytes 1 to n : path from MF (excluded)
LNG : length of identifier or of path
PATH : identifier or path

CLASS	INS	DATA OUT	
$03	$09	REND (3)	FCI(x)

REND : execution report
FCI : FCI

$03 $0A

Devices: *all printer models with ASK module*

Name: UPDATE RECORD (CD97)
Description : Deletion then writing of a record given in a linear fixed structure EF or the most recent recording of a circular file.

CLASS	INS	DATA IN				
$03	$0A	ACCES (1)	SID (1)	NREC (1)	LNG (1)	DATA (x)

ACCES : access mode
SID : SID (SID = $00 for usually selected EF)
NREC : recording number
LNG : number of bytes to write (n bytes £ size of a recording)
DATA : data to record

CLASS	INS	DATA OUT
$03	$0A	REND (3)

REND : execution report

$03 $0B

Devices: *all printer models with ASK module*

Name: VERIFY PIN (CD97)
Description : Presentation of PIN with counting of number of incorrect presentations.
A SelectFile of the MF should be carried out before being able to check PIN.
This command can't be used during a session.

Data In:

CLASS	INS	DATA IN					
$03	**$0B**	**MODE (1)**	**PIN (4)**	**TYPE_CMD (1)**	**NKEY/KIF (1)**	**00/KVC (1)**	**NSAM (1)**

MODE : $00, consultation of counter of number of incorrect presentations
$01, presentation of PIN
$02, presentation of PIN in transparent mode for contact communication
PIN : PIN (4 bytes)
TYPE_CMD: $00, short command (compatibility with the former one)
$01, long command
NKEY/ KIF: SAM key number to use
Or KIF of the key.
00/ KVC: $00 if NKEY passed in the previous parameter
or KVC of the Key.
NSAM : SAM number
$00 : default SAM,
$01, $02, $03 or $04 : logical number of the wanted SAM number

Data Out:

CLASS	INS	DATA OUT
$03	**$0B**	**REND (3)**

REND : execution report

$03 $0C

Devices: *all printer models with ASK module*

Name: WRITE RECORD (CD97)
Description : Writing without deletion of a record given in a linear fixed structure EF or the most recent recording of a circular file.

CLASS	INS	DATA IN				
$03	$0C	ACCES (1)	SID (1)	NREC (1)	LNG (1)	DATA (x)

ACCES : access mode
SID : SID (SID = $00 for usually selected EF)
NREC : recording number ($01 for a circular file)
LNG : number of bytes to write (n bytes £ size of a recording)
DATA : data to record

CLASS	INS	DATA OUT
$03	$0C	REND (3)

REND : execution report

$03 $0E

Devices: *all printer models with ASK module*

Name: GET ELECTRONIC PURSE STATUS (CD97)
Description : Informs on the status of PME and prepares purchase or loading, or purchase cancellation.

CLASS	INS	DATA IN
$03	$0E	TYPE (1)

TYPE : type of transaction to carry out
$00 : loading transaction
$01 : purchase transaction
$02 : purchase cancellation

- If loading transaction :

CLASS	INS	DATA OUT		
$03	$0E	REND (3)	PME (3)	JCHAR (22)

REND : execution report
PME : PME balance (Most significant Byte first)
JCHAR : 22 first bytes of most recent recording in loading journal

Date (2)
Money Batch (2)
Equipment Type (1)
Balance After Reloading (3)
Amount (3)
Time (2)
Security Device ID (4)
Security Device Transaction Number (3)
Tag Transaction Number (2)

- If purchase transaction or purchase cancellation :

CLASS	INS	DATA OUT		
$03	$0E	REND (3)	PME (3)	JPAID (19)

REND : execution report
PME : PME balance (Most significant Byte first)
JPAID : 19 first bytes of most recent recording in payments journal.

Amount (2)
Date (2)
Time (2)
Equipment Type (1)
Security Device ID (4)
Security Device Transaction Number (3)
Balance After Purchase (3)
Tag Transaction Number (2)

$03 $0D

Devices: *all printer models with ASK module*

Name: PURCHASE (CD97)
Description : Purchase with PME

CLASS	INS	DATA IN		
$03	$0D	ACHAT (1)	JPAID (7)	AFF (6)

ACHAT : type of purchase:
$00 : purchase transaction.
$01 : purchase transaction with display according to type of purchase.

JPAID : 7 first bytes of new recording in payments journal

Amount (2 bytes) : hexadecimal value of amount to debit from the card (Remark : this value should be negative).
Date (2 bytes) : Hexadecimal value of debit date. The debit date is usually equal to the number of days since 1/1/97.
Time (2 bytes) : Hexadecimal value of debit time. This time is usually equal to the number of minutes since 00h00m.
Equipment Type (1 bytes) : Hexadecimal value of type of equipment used to make the debit.

Example for a debit of 3 units on 09/12/98 at 15h29 :
Amount : $FF $FD
Date : $02 $C3
Time : $03 $A1
Equipment Type : $03

- If purchase transaction extended with display

AFF : 6 bytes of display (cf. display file).

CLASS	INS	DATA OUT
$03	$0C	REND (3)

REND : execution report
Remark: in an uninitialized card note that you must do First a Null Purchase
Ex: 03 0E 01 0B 00
03 0D 00 00 00 00 00 00 00 02 (See CD97 specification)

$03 $13

Devices: *all printer models with ASK module*

Name: CANCEL PURCHASE (CD97)
Description : Cancellation of previous payment done with the PME.

CLASS	INS	DATA IN	
$03	**$13**	**$00**	**JPAID (7)**

JPAID: 7 first bytes of new recording in payments journal

Amount (2)
Date (2)
Time (2)
Equipment Type (1)

Example for the cancellation of a debit of 3 units on 09/12/98 at 15h29 (example PURCHASE) :
Amount : $00 $03
Date : $02 $C3
Time : $03 $A1
Equipment Type : $03

- Cancellation of a transaction with display

CLASS	INS	DATA IN		
$03	**$13**	**$01**	**JPAID (7)**	**AFF (6)**

JPAID: 7 first bytes of the new recording in payments journal

Amount (2)
Date (2)
Time (2)
Equipment Type (1)

AFF : 6 bytes of display (cf. display file)

CLASS	INS	DATA OUT
$03	**$13**	**REND (3)**

REND : execution report

$03 $0F

Devices: *all printer models with ASK module*

Name: RELOAD ELECTRONIC PURSE (CD97)
DDescription : Charge of PME

CLASS	INS	DATA IN	
$03	**$0F**	**CHARG1 (5)**	**CHARG2 (5)**

CHARG1 : 5 first bytes of new record in loading journal
(Date, Money Batch, Equipment Type)

Date (2 bytes) : Hexadecimal value of loading date. The loading date is usually equal to the number of days since 1/1/97.
Money Batch (2 bytes) : Hexadecimal value of type of changes loaded on the card (i.e. : Francs, Euros, Dollars, etc.…)
Equipment Type (1 bytes) : Hexadecimal value of type of equipment used to do the loading.

CHARG2 : 5 bytes, offset [$08..$13], of new record in loading journal
(Amount, Time)

Amount (3 bytes) : hexadecimal value of amount to credit on the card (Remark : this value should be positive).
Time (2 bytes) : Hexadecimal value of loading time. This time is usually equal to the number of minutes since 00h00m.

CLASS	INS	DATA OUT
$03	**$0C**	**REND (3)**

REND : execution report
Remark: in an uninitialized card note that you must do First a Null Reload
Ex: 03 0E 00 0A 00
03 0F 00 00 00 02 00 00 00 00 00 00 (See CD97 specification)

$03 $10

Devices: *all printer models with ASK module*

Name: OPEN SECURED SESSION (CD97)
Description : Opening of a certification session.
Returns DF paths (from MF excluded) from non-ratified applications and the data read in the indicated record and file.
Remark : this command should be preceded by a select file command in the directory concerned.

CLASS	INS	DATA IN						
$03	**$10**	**TYPE (1)**	**SID (1)**	**NREC (1)**	**TYPE_CMD (1)**	**NKEY/KIF (1)**	**00/KVC (1)**	**MODE (1)**

TYPE : Type of operation : $00 : Personalization, $01 : Reloading, $02 : Validation
SID : SID (SID = $00 for usually selected EF)
NREC : recording number to read
$00 : no recording to read
$01: reading of a number 1 or unique recording
TYPE_CMD: $00, short command (compatibility with the former one for CD97 and GTML)
$01, long command
NKEY/ KIF: SAM key number to use
Or KIF of the key.
00/ KVC: $00 if NKEY passed in the previous parameter
or KVC of the Key.
MODE : Mode of operation ($00 : simple mode, $01 : extended mode)

CLASS	INS	DATA OUT						
$03	**$0B**	**REND (3)**	**NBAPP (1)**	**PATH1 (2)**	**...**	**PATHn (2)**	**DATA (29)**	**KCV (1)**

REND : execution report
NBAPP: 2*n where n is the number of non-ratified applications
PATH1 : path of the first non-ratified application
....
PATHn: path of the n^{th} non-ratified application
DATA : if recording to read, 29 bytes of data
KVC : KVC in extended mode.

$03 $11

Devices: *all printer models with ASK module*

Name: CLOSE SECURED SESSION (CD97)
Description : Closure of certification session
Example :

For an Increase function carried out in session we obtain:
$00 $90$00 $04 $nnnnnn where $nnnnnn is the new value of the counter.

Data In:

CLASS	INS	DATA IN	
$03	$11	TYPE_CMD (1)	TIMEOUT (1)

TYPE_CMD: $00, session will be ratified at the reception of the following command
$80, session is ratified immediately (except for CD97 and GTML)
$4A, switches OFF the field if the card doesn't answer.
TIMEOUT: if TYPE=$4A

Data Out:

CLASS	INS	DATA OUT	
$03	$11	REND (3)	RESULT (x)

REND : execution report
RESULT: result of incoming/outgoing orders during the session
The format is as follows :
1st byte : function number
following bytes: same format as for a function execution outside session apart from report (3 first bytes).

$03 $12

Devices: *all printer models with ASK module*

Name: CLOSE SECURED SESSION (CD97)
Description : Closure of certification session

CLASS	INS	DATA IN
$03	$12	-

No associated input data

CLASS	INS	DATA OUT
$03	$12	REND (3)

REND : execution report

$03 $15

Devices: *all printer models with ASK module*

Name: SELECT ISO APPLICATION
Description : Select application using Select File ISO command.

CLASS	INS	DATA IN		
$03	$15	Select Option (1)	LNG (1)	Data (x)

Select Option : 00 : first application or select by name if LNG <> 0.
01 : select last application (LNG should be 0)
02 : select next application (LNG should be 0)
03 : select previoust application (LNG should be 0)
LNG : length of data. 0 if Select Option <> 0, otherwise <= 16.
DATA : application name (LNG bytes)

CLASS	INS	DATA OUT	
$03	$12	REND (3)	FCI (x)

REND : execution report
FCI : FCI

3.7.5 Certificate Class (N°= $04)

$04 $04

Devices: *all printer models with ASK module*

Name: CheckCertificate (CTx)
Description : Certificate check

CLASS	INS	DATA IN					
$04	$04	KeyType	Param	LNG	BUFFER (x)	NB	Certificate (x)

KeyType: key type or number
Param: type of algorithm
LNG: buffer length (diversifier + data)
BUFFER: data to check (minimum 12 bytes = 8 bytes diversifier + 4 bytes data)
NB Certificate length (2 or 4 bytes)
Certificate(x): Certificate read in the CTX

CLASS	INS	DATA OUT		
$04	$04	Status	DATA1	DATA2

STATUS: 0x00: bad certificate
0x02: good certificate
0x03: SAM error
0x04: No SAM answer
DATA1 and DATA2 correspond to a SAM error code and are present only if the status is set to 0x03.

$04 $05

Devices: *all printer models with ASK module*

Name: giveCertificate (CTX)
Description : certificate generation

CLASS	INS	DATA IN				
$04	$05	KeyType	Param	LNG	BUFFER (x)	NB

KeyType: Key type or number
Param: type of algorithm
LNG: buffer length (diversifier + data)
BUFFER: data (minimum 12 bytes = 8 bytes diversifier + 4 bytes data)
NB Certificate length (2 or 4 bytes)

CLASS	INS	DATA OUT	
$04	$04	Status	Certificate (x)

STATUS: 0x00: bad certificate
0x02: good certificate
Certificate(x): Certificate calculated

3.7.6 Variable Class Mapping (N°= $05)

Created to enable use with identical commands for
- cards with fixed mapping: CD97 RJJ , GTML, GTML2
- cards with variable mapping : C97 RJL, CT2000

3.7.6.1 Main functions

The CSC Gen2xx coupler handled cards while taking into account a fixed structure of files (case of function classes CD97 (03) and GTML (02)) using a mapping table of the position of keys in the SAM according to their use.
This had the drawback of not being adaptable to new file structures and not being capable of handling several key versions.

So as to be able to handle the new variable file structure cards of the CD97 RJL and CT2000 types, handling by mapping table has been abandoned in this function class. To afford more scope for development of the applications (without changing coupler software), a "NKEY" parameter has been added to commands enabling the application to attribute the different SAM keys to the different card functions, in compliance with the choices retained in Card/SAM personalization.

The user can either communicate with the SAM thanks to a Key Reference or thanks to the KIF (indication of key function) and the KVC (indication of key version).

The error codes restituted by the card and the SAM are not filtered by the CSC and are consequently restituted wholly to the host.

For more information please refer to documents detailing default card and SAM mapping retained for the personalization of cards and SAMs.

3.7.6.2 Rules

Access mode :

value on 1 byte defining the principles of authentication used to access data.

DEFAULT	$00	No local cryptogram for the function. If a session is open, it is the session security that is used. Otherwise only the data of which the attribute is "ALWAYS" will be accessible.
PROTECTED	$01	Ground authentication, of incoming data for the badge. Immediate execution by the badge.
STAMPED	$02	Authentication of the badge, of outgoing data for the badge.

Remark : it is impossible to combine the mode "protected" or the mode "stamped" with the session mode.

3.7.6.3 Set of instruction

$05 $01

Devices: *all printer models with ASK module*

Name: APPEND RECORD
Description : Adds a record to an elementary circular file. This new record becomes the first in the file, the old one is suppressed.

CLASS	INS	DATA IN						
$05	**$01**	**ACCES (1)**	**SID (1)**	**LNG (1)**	**DATA (LNG)**	**LID (2)**	**NKEY/ KIF (1)**	**00/ KVC (1)**

ACCES : access mode
SID : SID (SID = $00 for usually selected EF)
LNG : length of data to record (n bytes £ size of a recording)
DATA : data to record
LID : LID (used in PROTECTED mode)
NKEY / KIF: SAM key number to use or KIF of the Key (used in PROTECTED mode)
00 / KVC: 00 if NKEY passed in the previous parameter or KVC of the Key (used in PROTECTED mode).

CLASS	INS	DATA OUT
$05	**$01**	**REND (3)**

REND : execution report

$05 $02

Devices: *all printer models with ASK module*

Name: CHANGE KEY

WARNING! This is a very specific command for personalization.

This command can't be used during a session.

CLASS	INS	DATA IN	
$05	**$02**	**$05**	**...**

$05 $02 $04

Devices: *all printer models with ASK module*

Name: CHANGE PIN
Description : Changes PIN value contained in the MF.
A SelectFile of the MF should be carried out before being able to change the PIN.
This command can't be used during a session.

Data In:

CLASS	INS	DATA IN						
$05	**$02**	**IKEY $04**	**KEY_NUM (1)**	**OLDPIN (4)**	**NEWPIN (4)**	**NKEY/ KIF (1)**	**00/ KVC (1)**	**ALG (1)**

IKEY : $04
KEY_NUM: key number ($00 for CD97, GTML and CT2000,
$04 for GTML2, CD21, CD Light
$09 for POPEYE)
OLDPIN : old PIN (used only for CD97)
NEWPIN : new PIN
NKEY / KIF: SAM key number to use or KIF of the Key
00 / KVC: 00 if NKEY passed in the previous parameter or KVC of the Key.
ALG: algorithm of the SAM used (for POPEYE card).

Data Out:

CLASS	INS	DATA OUT
$05	**$04**	**REND (3)**

REND : execution report

$05 $03

Devices: *all printer models with ASK module*

Name: DECREASE
Description : Decreases the value contained in a counter file.
Records the associated data.
Remark : Executed in session, this function can return or not (depending on the card type) the new value of the counter; for the CD97 card the new value will be returned by the CloseSecuredSession

CLASS	INS	DATA IN						
$05	$03	ACCES (1)	SID (1)	VALUE (3)	LID (2)	Index counter	NKEY/ KIF (1)	00/ KVC (1)

ACCES : access mode (default, protected)
SID : SID (SID = $00 for usually selected EF)
VALUE : value to deduct (3 bytes, binary number positive or nil)
LID : LID (used in PROTECTED mode)
Index counter : index of the counter (used in PROTECTED mode)
NKEY / KIF : SAM key number to use
or KIF of the Key (used in PROTECTED mode)
00 / KVC : $00 if NKEY passed in the previous parameter
or KVC of the Key (used in PROTECTED mode).

CLASS	INS	DATA OUT	
$05	$03	REND (3)	VALUE (3)

REND : execution report
VALUE: outside session, for all cards, new value (3 bytes, binary number signed) in session, for all cards except CD97, new value (3 bytes, binary number signed)

$05 $04

Devices: *all printer models with ASK module*

Name: INCREASE
Description : Increases the value contained in a counter file. Records the associated data.
Remark : Executed in session, this function can return or not (depending on the card type) the new value of the counter; for the CD97 card the new value will be returned by the CloseSecuredSession

CLASS	INS	DATA IN						
$05	$04	ACCES (1)	SID (1)	VALUE (3)	LID (2)	Index counter	NKEY/ KIF (1)	00/ KVC (1)

ACCES : access mode (default, protected)
SID : SID (SID = $00 for usually selected EF)
VALUE : value to add (3 bytes, binary number positive or nil)
LID : LID (used in PROTECTED mode)
Index counter : index of the counter (used in PROTECTED mode)
NKEY / KIF : SAM key number to use
or KIF of the Key (used in PROTECTED mode)
00 / KVC : $00 if NKEY passed in the previous parameter
or KVC of the Key (used in PROTECTED mode).

CLASS	INS	DATA OUT	
$05	$04	REND (3)	VALUE (3)

REND : execution report
VALUE: outside session, for all cards, new value (3 bytes, binary number signed) in session, for all cards except CD97, new value (3 bytes, binary number signed)

$05 $05

Devices: *all printer models with ASK module*

Name: INVALIDATE
Description : Invalidates the current DF (and consequently all offspring files).

CLASS	INS	DATA IN			
$05	$05	ACCES (1)	LID (2)	NKEY/ KIF (1)	00/ KVC (1)

ACCES : access mode
LID : LID (used in PROTECTED mode)
NKEY / KIF : SAM key number to use
or KIF of the Key (used in PROTECTED mode)
00 / KVC : $00 if NKEY passed in the previous parameter
or KVC of the Key (used in PROTECTED mode).

CLASS	INS	DATA OUT
$05	$05	REND (3)

REND : execution report

$05 $07

Devices: *all printer models with ASK module*

Name: REHABILITATE
Description : Cancels file invalidation.

CLASS	INS	DATA IN			
$05	$07	ACCES (1)	LID (2)	NKEY/ KIF (1)	00/ KVC (1)

ACCES : access mode
LID : LID (used in PROTECTED mode)
NKEY / KIF : SAM key number to use
or KIF of the Key (used in PROTECTED mode)
00 / KVC : $00 if NKEY passed in the previous parameter
or KVC of the Key (used in PROTECTED mode).

CLASS	INS	DATA OUT
$05	$05	REND (3)

REND : execution report

$05 $06

Devices: *all printer models with ASK module*

Name: READ RECORD
Description : Reading of a record given in a circular EF, a counter or a linear fixed structure EF.

CLASS	INS	DATA IN						
$05	$06	ACCES (1)	SID (1)	NREC (1)	LNG (1)	LID (2)	NKEY/ KIF (1)	00/ KVC (1)

ACCES : access mode
SID : SID (SID = $00 for usually selected EF)
NREC : recording number
LNG : length of data to read
LID : LID (used in STAMPED mode)
NKEY / KIF: SAM key number to use
or KIF of the Key (used in STAMPED mode)
00 / KVC: $00 if NKEY passed in the previous parameter
or KVC of the Key (used in STAMPED mode).

CLASS	INS	DATA OUT	
$05	$06	REND (3)	DATA (x)

REND : execution report
DATA : data read (n bytes)

$05 $08

Devices: *all printer models with ASK module*

Name: SELECT FILE
Description : Description : Explicit selection of current EF or DF.
This command sends back the file description data.

CLASS	INS	DATA IN		
$05	$08	CNTR (1)	LNG (1)	PATH (x)

CNTR : Selection check
$00, MF
$02, EF in the current DF (identifier bytes 2 to n)
$08, bytes 1 to n : path from MF (excluded)
LNG : length of identifier or of path
PATH : identifier or path

NB: The command $05 $08 $00 $00 is not supported by CD97 card.

CLASS	INS	DATA OUT	
$05	$08	REND (3)	FCI (x)

REND : execution report
FCI : FCI

$05 $09

Devices: *all printer models with ASK module*

Name: STATUS
Description : ditto SELECT FILE, but without selecting a file. (not available on any card)

CLASS	INS	DATA IN		
$05	$09	CNTR (1)	LNG (1)	PATH (x)

CNTR : Selection check
$00, MF
$02, EF in the current DF (identifier bytes 2 to n)
$08, bytes 1 to n : path from MF (excluded)
LNG : length of identifier or of path
PATH : identifier or path

CLASS	INS	DATA OUT	
$05	$09	REND (3)	FCI (x)

REND : execution report
FCI : FCI

$05 $0A

Devices: *all printer models with ASK module*

Name: UPDATE RECORD
Description : Deletion then writing of a record given in a linear fixed structure EF or the most recent recording of a circular file.

CLASS	INS	DATA IN							
$05	**$0A**	**ACCES (1)**	**SID (1)**	**NREC (1)**	**LNG (1)**	**DATA (x)**	**LID (2)**	**NKEY/ KIF (1)**	**00/ KVC (1)**

ACCES : access mode
SID : SID (SID = $00 for usually selected EF)
NREC : recording number
LNG : number of bytes to write (n bytes ≤ size of a recording)
DATA : data to record
LID : LID (used in PROTECTED mode)
NKEY / KIF: SAM key number to use or KIF of the Key (used in PROTECTED mode)
00 / KVC: 00 if NKEY passed in the previous parameter or KVC of the Key (used in PROTECTED mode).

CLASS	INS	DATA OUT
$05	**$0A**	**REND (3)**

REND : execution report

$05 $0B

Devices: *all printer models with ASK module*

Name: VERIFY PIN
Description : Presentation of PIN with counting of number of incorrect presentations.
A SelectFile of the MF should be carried out before being able to check PIN.
This command can't be used during a session.

CLASS	INS	DATA IN			
$05	$0B	MODE (1)	PIN (4)	NKEY/ KIF (1)	00/ KVC (1)

MODE : $00, consultation of the counter of incorrect presentations
$01, presentation of PIN in encrypted mode for RF communication
$02, presentation of PIN in transparent mode for contact communication
PIN : PIN (4 bytes)
NKEY / KIF : SAM key number to use or KIF of the Key
00 / KVC : 00 if NKEY passed in the previous parameter or KVC of the Key.

CLASS	INS	DATA OUT
$05	$0B	REND (3)

REND : execution report

$05 $0C

Devices: *all printer models with ASK module*

Name: WRITE RECORD
Description : Writing without deletion of a record given in a linear fixed structure EF or the most recent recording of a circular file.

CLASS	INS	DATA IN							
$05	$0C	ACCES (1)	SID (1)	NREC (1)	LNG (1)	DATA (x)	LID (2)	NKEY/ KIF (1)	00/ KVC (1)

ACCES : access mode
SID : SID (SID = $00 for usually selected EF)
NREC : recording number ($01 for a circular file)
LNG : number of bytes to write (n bytes £ size of a recording)
DATA : data to record
LID : LID (used in PROTECTED mode)
NKEY / KIF: SAM key number to use
or KIF of the Key (used in PROTECTED mode)
00 / KVC: $00 if NKEY passed in the previous parameter
or KVC of the Key (used in PROTECTED mode).

CLASS	INS	DATA OUT
$05	$0C	REND (3)

REND : execution report

$05 $10

Devices: *all printer models with ASK module*

Name: OPEN SECURED SESSION
Description : Opening of a certification session.
Returns DF paths (from MF excluded) of non-ratified applications and data read in the indicated record and file.
Remark: this command should be preceded by a select file command in the directory concerned.

CLASS	INS	DATA IN					
$05	**$10**	**TYPE (1)**	**SID (1)**	**NREC (1)**	**NKEY/ KIF (1)**	**00/ KVC (1)**	**MODE (1)**

TYPE : Type of operation : $00 : Personalization, $01 : Reloading, $02 : Validation
SID : SID (SID = $00 for usually selected EF)
NREC : recording number to read
$00 : no recording to read
$01: reading of a number 1 or unique recording
NKEY / KIF : SAM key number to use or KIF of the Key
00 / KVC or FF : * 00 if NKEY passed in the previous parameter
* or KVC of the Key.
* or 0xFF if the KVC is unknown.
MODE : Mode of operation ($00 : simple mode, $01 : extended mode:the KVC used is sent back)
The extended mode is not supported by CD97 and GTML .

CLASS	INS	DATA OUT						
$05	**$10**	**REND (3)**	**NBAPP (1)**	**PATH1 (2)**	**...**	**PATHn (2)**	**DATA (29)**	**KVC (1)**

REND : execution report
NBAPP: 2*n where n is the number of non-ratified applications
PATH1 : path of the first non-ratified application
....
PATHn: path of the n^{th} non-ratified application
DATA : if recording to read, 29 bytes of data
KVC : KVC in extended mode.

$05 $11

Devices: *all printer models with ASK module*

Name: CLOSE SECURED SESSION
Description : Closure of certification session
Example: For an Increase function executed in session we obtain:
$00 $90$00 $04 $nnnnnn where $nnnnnn is the new value of the counter.

CLASS	INS	DATA IN	
$05	$11	TYPE_CMD (1)	TIMEOUT (1)

TYPE_CMD: $00, session will be ratified at the reception of the following command
$80, session is ratified immediately (except for CD97 and GTML1)
$4A, switches OFF the field if the card doesn't answer.
TIMEOUT: if TYPE=$4A

CLASS	INS	DATA OUT	
$05	$11	REND (3)	RESULT (x)

REND : execution report
RESULT: result of incoming/outgoing orders during the session
The format is as follows :
1er byte : function number
following bytes: same format as for a function execution outside session apart from the report (3 first bytes).

$05 $12

Devices: *all printer models with ASK module*

Name: ABORT SECURED SESSION
Description : Stop the current certification session. This still allow to continue to dialogue with the badge and in particular, open a new session.

CLASS	INS	DATA IN
$05	$12	-

No associated input data

CLASS	INS	DATA OUT
$05	$12	REND (3)

REND : execution report

$05 $14

Devices: *all printer models with ASK module*

Name: DECREASE MULTIPLE
Description : Decrease the value contained in several counters.
Remark : This function should be carried out in session and it immediately returns the new value of the counters.
This command is not supported by CD97 and GTML .

CLASS	INS	DATA IN						[	NBC fois		]
$05	$14	ACCES (1)	SID (1)	LID (2)	NKEY / KIF(1)	00/ KVC (1)	NBC(1)	[	CI (1)	VALUE (3)	]

ACCES : access mode.
SID : Short ID
NKEY / KIF : SAM key number to use or KIF of the Key
00 / KVC : 00 if NKEY passed in the previous parameter or KVC of the Key.
NBC : Number of counters concerned (maximum 7)
CI : Counter index
VALUE : value to deduct (3 bytes, binary number positive or nil)

CLASS	INS	DATA OUT	[	NBC fois		]
$05	$14	REND (3)	[	CI (1)	VALUE (3)	]

REND : execution report
CI : Counter index
VALUE: new value (3 bytes, binary number signed)

$05 $15

Devices: *all printer models with ASK module*

Name: INCREASE MULTIPLE
Description : Increases the value contained in a counter file.
The associated data is not written in the GTML.
Remark : This function should be carried out in session and it immediately returns the new value of the counter.
This command is not supported by CD97 and GTML .

CLASS	INS	DATA IN							NBC fois	
$05	$15	ACCES (1)	SID (1)	LID (2)	NKEY / KIF(1)	00/ KVC (1)	NBC(1)	[	CI (1)	VALUE (3)]

ACCES : access mode
SID : SID (SID = $00 for usually selected EF)
NKEY / KIF : SAM key number to use or KIF of the Key
00 / KVC : 00 if NKEY passed in the previous parameter or KVC of the Key.
NBC : Number of counters concerned (maximum 7)
CI : Counter index
VALUE: value to add (3 bytes, binary number positive or nil)

CLASS	INS	DATA OUT		NBC fois	
$05	$15	REND (3)	[	CI (1)	VALUE (3)]

REND : execution report
CI : Counter index
VALUE: new value (3 bytes, binary number signed)

$05 $12

Devices: *all printer models with ASK module*

Name: LOCK / UNLOCK
Description : Locks / Unlocks the card.
REMARQUE : Before this command, a Verify Pin must be carried out,
After this command, the current file is unchanged.

CLASS	INS	DATA IN
$05	$16	CMD (1)

CMD : Type of Command : LOCK = $00 / UNLOCK = $01

CLASS	INS	DATA OUT
$05	$16	REND (3)

REND : execution report

$05 $0D

Devices: *all printer models with ASK module*

Name: PURCHASE
Description : Purchase with PME
Remark : maintained for compatibility with the CD97

CLASS	INS	DATA IN		
$05	$0D	ACHAT (1)	JPAID (7)	AFF (6)

ACHAT : Type of purchase : $00 : purchase transaction.
$01 : purchase transaction with display according to type of purchase.
JPAID : 7 1st bytes of new recording in payments journal (Amount, Date, Time, Equipment Type)
NB: the Amount field (2 bytes) has to be specified in negative value.

- If extended purchase transaction with display

AFF : 6 bytes of display (cf. display file).

CLASS	INS	DATA OUT
$05	$0D	REND (3)

REND : execution report
Remark: in an uninitialized card note that you must do First a Null Purchase
Ex: 05 0E 01 0B 00
05 0D 00 00 00 00 00 00 00 02 (See CD97 specification)

$05 $0E

Devices: *all printer models with ASK module*

Name: GET ELECTRONIC PURSE STATUS
Description : Informs on PME status and prepares purchase or loading or a purchase cancellation.
Remark : maintained for compatibility with the CD97

CLASS	INS	DATA IN		
$05	$0E	TYPE (1)	NKEY (7)	RUF (1)

TYPE : type of transaction to do
$00 : loading transaction
$01 : purchase transaction
$02 : purchase cancellation
NKEY: Key Number to use

- If loading transaction :

CLASS	INS	DATA OUT			
$05	$0E	REND (3)	PME (3)	JCHAR (22)	JINFO(6)

REND : execution report
PME : PME balance (Most significant Byte first)
JCHAR : 22 first bytes of most recent recording in loading journal (Date, Money Batch, Equipment Type,Balance After Reloading, Amount, Time, Security Device ID, Security Device Transaction, Number, Tag Transaction Number)
JINFO: 6 bytes of information (Current KVC, Current Card Transaction Number, Previous CryptoLo)

- If purchase transaction or purchase cancellation :

CLASS	INS	DATA OUT			
$05	$0E	REND (3)	PME (3)	JPAID (19)	JINFO(6)

REND : Execution report
PME : PME balance (Most significant Byte first)
JPAID : 19 first bytes of most recent recording in payments journal . (Amount, Date, Time, Equipment Type, Security Device ID, Security Device Transaction Number, Balance After Purchase, Tag Transaction Number) JINFO: 6 bytes of information (Current KVC, Current Card Transaction Number, Previous CryptoLo)

$05 $0F

Devices: *all printer models with ASK module*

Name: RELOAD ELECTRONIC PURSE
Description : PME loading
Remark : maintained for compatibility with the CD97

CLASS	INS	DATA IN	
$05	$0F	CHARG1 (5)	CHARG2 (5)

CHARG1 : 5 first bytes of new recording in loading journal (Date, Money Batch, Equipment Type) See CD97.
CHARG2 : 5 bytes, offset [$08..$13], of new recording in loading journal (Amount, Time) See CD97.

CLASS	INS	DATA OUT
$05	$0F	REND (3)

REND : execution report
Remark: in an un-initialized card note that you must do First a Null Reload
Ex: 05 0E 00 0A 00
05 0F 00 00 00 02 00 00 00 00 00 00 (See CD97 specification)

$05 $13

Devices: *all printer models with ASK module*

Name: CANCEL PURCHASE
Description : Cancellation of previous payment carried out with the PME.
Remark : maintained for compatibility with the CD97

- Cancellation of a transaction without display

CLASS	INS	DATA IN	
$05	$13	$00	JPAID (7)

JPAID: 7 1st bytes of new recording in payments journal (Amount, Date, Time, Equipment Type)

- Cancellation of a transaction with display

CLASS	INS	DATA IN		
$05	$13	$01	JPAID (7)	AFF (6)

JPAID: 7 1st bytes of new recording in payments journal (Amount, Date, Time, Equipment Type)
AFF : 6 bytes of display (cf. display file)
NKEY : SAM key number to use

CLASS	INS	DATA OUT
$05	$13	REND (3)

REND : execution report

$05 $23

Devices: *all printer models with ASK module*

Name: DECREASE_LG
Description : It is a command for CD97 card only,
Decreases the value contained in a counter file and writes the 5 free data. Records the associated data.
Remark : Executed in session, the new value of the counter will be returned by the CloseSecuredSession

Data In:

CLASS	INS	DATA IN						
$05	**$23**	**ACCES (1)**	**SID (1)**	**VALUE (8)**	**LID (2)**	**Index counter (1)**	**NKEY/KIF (1)**	**00/KVC (1)**

ACCES : access mode (default, protected)
SID : SID (SID = $00 for usually selected EF)
VALUE : value to deduct (3 bytes, binary number positive or nil)
+ 5 free bytes.
LID : LID (used in PROTECTED mode)
Index counter : index of the counter (used in PROTECTED mode)
NKEY / KIF : SAM key number to use
or KIF of the Key (used in PROTECTED mode)
00 / KVC : $00 if NKEY passed in the previous parameter
or KVC of the Key (used in PROTECTED mode).

Data Out:

CLASS	INS	DATA OUT	
$05	**$23**	**REND (3)**	**VALUE (3)**

REND : execution report
VALUE: outside session, for all cards, new value (3 bytes, binary number signed) in session, nothing

$05 $24

Devices: *all printer models with ASK module*

Name: INCREASE_LG
Description : It is a command for CD97 card only,
Increases the value contained in a counter file and writes the 5 free data.
Records the associated data
Remark : Executed in session, the new value of the counter will be returned by the CloseSecuredSession

Data In:

CLASS	INS	DATA IN						
$05	$24	ACCES (1)	SID (1)	VALUE (8)	LID (2)	Index counter (1)	NKEY/KIF (1)	00/KVC (1)

ACCES : access mode (default, protected)
SID : SID (SID = $00 for usually selected EF)
VALUE : value to add (3 bytes, binary number positive or nil)
+ 5 free bytes.
LID : LID (used in PROTECTED mode)
Index counter : index of the counter (used in PROTECTED mode)
NKEY / KIF : SAM key number to use
or KIF of the Key (used in PROTECTED mode)
00 / KVC : $00 if NKEY passed in the previous parameter
or KVC of the Key (used in PROTECTED mode).

Data Out:

CLASS	INS	DATA OUT	
$05	$24	REND (3)	VALUE (3)

REND : execution report
VALUE: outside session, for all cards, new value (3 bytes, binary number signed) in session, nothing

3.7.7 CTS256B Class (N°= $06)

The CTS256B is one of the members of the range of ASK contactless tickets. It has a memory of 256 bits and is intended for applications where active authentication of the ticket is not necessary, for example ticket for immediate use etc ...

3.7.7.1 Memory organization

The CTS256B is addressed by words of 2 Bytes. The CSC handles Byte level.

EEPROM 16 words

Address	Area	Msb word lsb	
0	1	Byte 1	Byte 0
1	2	Byte 3	Byte 2
2	3	Byte 5	Byte 4
3		Byte 7	Byte 6
4		Byte 9	Byte 8
5	4	Byte 11	Byte 10
6		Byte 13	Byte 12
7		Byte 15	Byte 14
8		Byte 17	Byte 16
9		Byte 19	Byte 18
10		Byte 21	Byte 20
11	5	Byte 23	Byte 22
12		Byte 25	Byte 24
13	6	Byte 27	Byte 26
14		Byte 29	Byte 28
15		Byte 31	Byte 30

3.7.7.2 Set of instruction

$06 $01

Devices: *all printer models with ASK module*

Name: ACTIVE (CTS265B)
Description: Activates only the CTS ticket and sends back the first 5 blocks (Equivalent to EnterHuntPhase card).

CLASS	INS
$06	$01

CLASS	INS	DATA OUT		
$06	$01	Length	Status	DATA (x)

Length: number of response bytes
Status: $00: communication interrupted
$01: bad CRC
$0F: success
$40: detection of a card which is not a CTS 256
$80: collision
DATA: data read (Length -1 byte): maximum 8 bytes of series number

$06 $02

Devices: *all printer models with ASK module*

Name: READ (CTS256B)
Description: Reading of a number of bytes at a given address.

CLASS	INS	DATA IN	
$06	$02	ADD	NB

ADD: Address of the first reading (0…31), in bytes.
NB: Number of bytes to read, from 1 to 32

CLASS	INS	DATA OUT		
$06	$02	Length	Status	DATA (x)

Length: response length
Status: $00: communication interrupted
$01: bad CRC
$02: success
$03: invalid parameters
DATA: data read (Length -1 byte): maximum 8 bytes of series number

$06 $03

Devices: *all printer models with ASK module*

Name: UPDATE (CTS256B)
Description: Deletion if necessary, recording, then checks by reading bytes written at ADD address.

CLASS	INS	DATA IN			
$06	$03	ADD	NB	DATA (x)	DATAinCTS (x)

ADD: Address of 1st byte to record (0…31).
NB: Number of bytes to read, from 1 to 32
DATA: Data to record, NB bytes
DATAinCTS: Data already read and recorded in the ticket (NB bytes) enables quicker recording. If the data is not known or is required to be deleted, every byte of DATAinCTS must be set to 0xEE .

CLASS	INS	DATA OUT		
$06	$03	Length	Status	DATA (x)

Length: length of written and read data in bytes.
Status: $00: No response
$01: Bad CRC
$02: Success
$03: invalid parameters
$8x: Security activated (i.e. data written =/= data read)
$82: Security activated + good CRC
DATA: data read (Length -1 byte)

$06 $04

Devices: *all printer models with ASK module*

Name: RELEASE (CTS256B)
Description: Deactivation of the CTS.

CLASS	INS	DATA IN
$06	$04	Param

Param: 00 : deactivation of the ticket using the instruction “deactivate”
Others : RFU

CLASS	INS	DATA OUT
$06	$04	Status

Status: $00: ticket always active
$02: ticket deactivated

3.7.8 CTx512x Class (N°= $06)

3.7.8.1 CTx512B

The CTx512B is one of the members of the range of ASK contactless tickets. It has a memory of 512 bits and is intended for applications where anticollision between tickets is necessary.
CTM512B contains very high security features

The CTx512B is addressed by words of 2 bytes. The CSC handles byte level.

CTx512B internal EEPROM 64 bytes organisation and physical address:

Address	Area	Msb word lsb	
0	1	Byte 1	Byte 0
1	2	Byte 3	Byte 2
2	3	Byte 5	Byte 4
3		Byte 7	Byte 6
4		Byte 9	Byte 8
5	4	Byte 11	Byte 10
6		Byte 13	Byte 12
7		Byte 15	Byte 14
8		Byte 17	Byte 16
9		Byte 19	Byte 18
10	5	Byte 21	Byte 20
11		Byte 23	Byte 22
12		Byte 25	Byte 24
13		Byte 27	Byte 26
14		Byte 29	Byte 28
15		Byte 31	Byte 30

Address	Area	Msb word lsb	
16	5	Byte 33	Byte 32
17		Byte 35	Byte 34
18	6	Byte 37	Byte 36
19		Byte 39	Byte 38
20	7	Byte 41	Byte 40
21		Byte 43	Byte 42
22	8	Byte 45	Byte 44
23		Byte 47	Byte 46
24	9	Byte 49	Byte 48
25		Byte 51	Byte 50
26	10	Byte 53	Byte 52
27		Byte 55	Byte 54
28	11	Byte 57	Byte 56
29		Byte 59	Byte 58
30		Byte 61	Byte 60
31		Byte 63	Byte 62

3.7.8.2 Mifare® UltraLight

The Mifare® UltraLight is a member of the Mifare® family but is considered as a ticket whose characteristics are very close to the CTx521B's ones. That's why the functions of the Mifare® UltraLight belong to the same class as the CTx521B class. But this card has to be searched with an ISO A search and not a Mifare® search.

The communication layer of the UltraLight complies to parts 2 and 3 (but not 4!) of the ISO 14443 A standard. Anti-collision and Security features are also implemented.

The 512 bit EEPROM memory is addressed by 4-bytes wide pages, but the CSC handles byte level. It is organised in 16 pages:

BYTE NUMBER	0	1	2	3	Page
Serial Number	SN 0	SN 1	SN 2	BCC0	**0**
Serial Number	SN 3	SN 4	SN 5	SN 6	**1**
Internal / Lock	BCC1	Internal	Lock 0	Lock 1	**2**
OTP	OTP 0	OTP 1	OTP 2	OTP 3	**3**
DATA (read/write)	Data 0	Data 1	Data 2	Data 3	**4**
DATA (read/write)	Data 4	Data 5	Data 6	Data 7	**5**
DATA (read/write)	Data 8	Data 9	Data 10	Data 11	**6**
DATA (read/write)	Data 12	Data 13	Data 14	Data 15	**7**
DATA (read/write)	Data 16	Data 17	Data 18	Data 19	**8**
DATA (read/write)	Data 20	Data 21	Data 22	Data 23	**9**
DATA (read/write)	Data 24	Data 25	Data 26	Data 27	**10**
DATA (read/write)	Data 28	Data 29	Data 30	Data 31	**11**
DATA (read/write)	Data 32	Data 33	Data 34	Data 35	**12**
DATA (read/write)	Data 36	Data 37	Data 38	Data 39	**13**
DATA (read/write)	Data 40	Data 41	Data 42	Data 43	**14**
DATA (read/write)	Data 44	Data 45	Data 46	Data 47	**15**

3.7.8.3 Functions list

$06 $20

Devices: *all printer models with ASK module*

Name: LIST (CTx512B ONLY)
Description: List performs anticollision and answers the serial numbers of all the CTx512B present in the reader field
After LIST instruction, all the tickets are in HALT state. Each ticket has to be selected by its serial number before any other command.

After 3 REQT without answer, LIST instruction answers Length = $02, status = $00 and NB = $00

If the Field off CTx configuration is active, the field is turned on and a select is done before the command, and the field is turned off after the command.

CLASS	INS	RFU
$06	$20	$00

CLASS	INS	DATA OUT			
$06	$20	Length	Status	NB	SERIAL (x)

Length: number of response bytes (Length = $02 + 2 x NB)
status $00 no ticket
$x1 CTx512B in antenna field
$x2 CTS256B in antenna field
$x3 CTx512B and CTS256B in antenna field
$x4 identification error (chip version or manufacturer)
$x5 mysterious answer
$8x Time out reached before end of anti-collision: problem occurs
NB: $00 no ticket
$xx: number of tickets which are responding in the field
SERIAL: list of serial data: composed by the 2 LSB serial number (address $03).

$06 $21

Devices: *all printer models with ASK module*

Name: SELECT (CTx512B ONLY)
Description: Selects a specific ticket by this serial number

If the Field off CTx configuration is active, the field is turned on and a select is done before the command, and the field is turned off after the command.

CLASS	INS	DATA IN	
$06	**$21**	**serial**	**serial**

SERIAL: Serial number (two LSBytes)

CLASS	INS	DATA OUT		
$06	**$21**	**Length**	**Status**	**DATA (x)**

Length: answer length
Status: $00: No answer
$01: Bad CRC
$02: Success
DATA: Serial number read (should be same as serial)

$06 $22

Devices: *all printer models with ASK module*

Name: READ (CTx512B and MIFARE® ULTRALIGHT)
Description: Reading of a number of bytes at a given address.
Internally, the reader chooses read or multi-read instruction depending on NB parameter.

If the Field off CTx configuration is active, the field is turned on and a select is done before the command, and the field is turned off after the command (only in CTx512b mode).

CLASS	INS	DATA IN	
$06	**$22**	**Byte Number**	**NB**

Byte Number: Address of the first reading (0…63), in bytes.
NB: Number of bytes to read, from 1 to 64

CLASS	INS	DATA OUT		
$06	**$22**	**Length**	**Status**	**DATA (x)**

Length: response length
Status: $00: No answer
$01: Bad CRC
$02: Success
$03: Bad Parameters
Note: For the Mifare® UltraLight if Status is different from $02 or $03 the card will come into the HALT state, so you have to wake it up to perform other transactions.
DATA: data read (Length -1 byte).

$06 $23

Devices: *all printer models with ASK module*

Name: UPDATE (CTx512B and MIFARE® ULTRALIGHT)
Description: Deletes, records, then checks by reading the bytes written at ADD address.

If the Field off CTx configuration is active, the field is turned on and a select is done before the command, and the field is turned off after the command (only in CTx512b mode).

CLASS	INS	DATA IN		
$06	$23	Byte Number	NB	DATA (x)

Byte Number: Address of 1st byte to record (0…63)
NB: Number of bytes to update, from 1 to 64
DATA: Data to be updated, NB bytes

CLASS	INS	DATA OUT		
$06	$23	Length	Status	DATA (x)

Length: length of written and read data in bytes.
Status: $00 : No answer
$01: Bad CRC
$02: Success
$03: Bad parameters
$82: Security activated
Note: For the Mifare® UltraLight if Status is different from $02 or $03 the card will come into the HALT state, so you have to wake it up to perform other transactions.
DATA: data read after a successful write (Length -1 byte)

NB for the Mifare® UltraLight exclusively: due to the checking of the written data, if the writing phase has not completed successfully, the Mifare®UltraLight will goes into the HALT state and so, a TimeOut will be sent back.

$06 $24

Devices: *all printer models with ASK module*

Name: HALT (CTx512B and MIFARE® ULTRALIGHT)
Description: Halt CTx512x ticket.

If the Field off CTx configuration is active, the field is turned on and a select is done before the command, and the field is turned off after the command (only in CTx512b mode).

CLASS	INS	DATA IN
$06	$24	Param

Param: 00 : deactivation of the ticket using the instruction “deactivate”
Others : RFU

CLASS	INS	DATA OUT
$06	$24	Status

Status: $00: Ticket still active
$02: Ticket deactivated

$06 $26

Devices: *all printer models with ASK module*

Name: WRITE (CTx512B and MIFARE® ULTRALIGHT)
Description: Performs a true “Write” operation: puts bits to “1” if not but cannot write a 0 instead of a 1. It is useful for writing in OTP area or eventually to use the whole Data area as an OTP zone.

If the Field off CTx configuration is active, the field is turned on and a select is done before the command, and the field is turned off after the command (only in CTx512b mode).

CLASS	INS	DATA IN		
$06	$26	Byte Number	NB	DATA (x)

Byte Number: Address of 1st byte to record (0…63)
NB: Number of bytes to write, from 1 to 64
DATA: Data to be written, NB bytes

CLASS	INS	DATA OUT		
$06	$26	Length	Status	DATA (x)

Length: length of written and read data in bytes.
Status:
$00: No answer
$01: Bad CRC
$02: Success
$03: Bad parameters
$82: Security activated
Note: For the Mifare® UltraLight if Status is different from $02 or $03 the card will come into the HALT state, so you have to wake it up to perform other transactions.
DATA: data obtained after the Write operation: the former existing data plus the written data computed with the “OR” operation (Length -1 byte)

$06 $27

Devices: *all printer models with ASK module*

Name: AUTHENTICATE (CTM512B ONLY)
Description: authenticates an area of the ticket (8 consecutive bytes), thanks to the Anticlone function of the SAM.

If the Field off CTx configuration is active, the field is turned on and a select is done before the command, and the field is turned off after the command (only in CTx512b mode).

CLASS	INS	DATA IN		
$06	**$27**	**Address**	**KIF / KeyRef**	**KVC / 0x00**

Address: Address of the area to authenticate
KIF / KeyRef: Specifies the KIF or the KeyRef, but if KeyRef, the following byte must be set to 0x00.
KVC / 0x00: Specifies the KVC if KIF has been specified before, if not must be 0x00.

CLASS	INS	DATA OUT		
$06	**$27**	**Length**	**Status**	**DATA (x)**

Length: Status length + Data length.
Status:
$00: No answer (TimeOut)
$01: Unexpected failure
$02: Success
$03: Bad parameters
$04: No current SAM
$05: SAM not initialized
$06: Bas Status SAM
DATA: present only to get the SAM status code back(only when STATUS = $06).

$06 $28

Devices: *all printer models with ASK module*

Name: WRITE KEY (CTM512B ONLY)
Description: After the personalization phase, writes a diversified key in the ticket.

If the Field off CTx configuration is active, the field is turned on and a select is done before the command, and the field is turned off after the command (only in CTx512b mode).

CLASS	INS	DATA IN	
$06	**$28**	**KIF / KeyRef**	**KVC / 0x00**

KIF / KeyRef: Specifies the KIF or the KeyRef, but if KeyRef, the following byte must be set to 0x00.
KVC / 0x00: Specifies the KVC if KIF has been specified before, if not must be 0x00.

CLASS	INS	DATA OUT		
$0	**$28**	**Length**	**Status**	**DATA (x)**

Length: Status length + Data length.
Status:
$00: No answer (TimeOut)
$01: Unexpected failure
$02: Success
$03: Bad parameters
$04: No current SAM
$05: SAM not initialized
$06: Bas Status SAM
$07: Writing prohibited
DATA: present only to get the SAM status code back(only when STATUS = $06).

$06 $29

Devices: *all printer models with ASK module*

Name: UPDATE_FIELD_ON (CTx512B and MIFARE® ULTRALIGHT)
Description: Deletes, records, then checks by reading the bytes written at ADD address.

If the Field off CTx configuration is active, the field is turned on and a select is done before the command, but is not turned off after the command (only in CTx512b mode).

CLASS	INS	DATA IN		
$06	$29	Byte Number	NB	DATA (x)

Byte Number: Address of 1st byte to record (0…63)
NB: Number of bytes to write, from 1 to 64
DATA: Data to be updates, NB bytes

CLASS	INS	DATA OUT		
$0	$29	Length	Status	DATA (x)

Length: length of written and read data in bytes.
Status:
$00: No answer
$01: Bad CRC
$02: Success
$03: Bad parameters
$82: Security activated
Note: For the Mifare® UltraLight if Status is different from $02 or $03 the card will come into the HALT state, so you have to wake it up to perform other transactions.
DATA: data read after a successful write (Length -1 byte)

3.7.9 SR / SRI / SRT / SRIX Class (N°= $06)

The SR / SRI / SRT / SRIX are members of the range of ASK contactless tickets. They come with different memory organizations. The SRIX anti-clone functionality is not handled by the firmware as it is France Telecom proprietary algorithm.
These tickets are intended for applications where active authentication of the ticket is not necessary, for example ticket for immediate use etc ...

3.7.9.1 Memory organization

	SR176	SR512 family (SRI512 / SRT512 / SRIX512)	SR4K family (SRI4K/SRIX4K)
User EEPROM (bits)	176	512	4096
Mapping (blocks * bits)	11 x 16	16 x 32	128 x 32
UID (bits)	64	64	64

This chips family is addressed by blocks of 2 bytes (SR176) or 4 bytes (SR512 and SR4K families).

The CSC firmware handles byte level as well as the native blocks organization.

3.7.9.2 Set of instruction

$06 $31

Devices: *all printer models with ASK module*

Name: ACTIVE (SR Family)
Description: Activate and select a SR, SRI, SRT or SRIX ticket and send back the chip type and the 64-bit UID.

CLASS	INS
$06	$31

CLASS	INS	DATA OUT			
$06	$31	Length	Status	Chip type	UID

Length: number of response bytes (here $0A)
Status: $00: communication interrupted
$01: bad CRC
$0F: success
$80: collision
Chip type: $00: SR176
$01: SR512
$02: SR4K
UID: 64-bit (8-byte) UID from LSB to MSB (UID0, UID1… UID7)

$06 $32

Devices: *all printer models with ASK module*

Name: READ BLOCKS (SR Family)
Description: Read blocks.

CLASS	INS	DATA IN	
$06	$32	BLOCK	NB

BLOCK : Fist block number to read
SR176 : 0 to 15 ($00 to $0F)
SR512 family : 0 to 15 ($00 to $0F) or 255 ($FF)
SR4K family : 0 to 127 ($00 to $7F) or 255 ($FF)
NB: Number of blocks to read
SR176 : 1 to 16 ($01 to $10)
SR512 family : 1 to 16 ($01 to $10)
SR4K family : 1 to 60 ($01 to $3C)

Note : SR176 has 2-byte blocks, SR512 and SR4K families have 4-byte blocks.

CLASS	INS	DATA OUT		
$06	$32	Length	Status	DATA (x)

Length: response length
Status: $00: communication interrupted
$01: bad CRC
$02: success
$03: bad parameters
DATA: data read (Length -1 byte)

$06 $33

Devices: *all printer models with ASK module*

Name: WRITE BLOCKS (SR Family)
Description: Write and verify blocks.

CLASS	INS	DATA IN		
$06	$33	BLOCK	NB	DATA (x)

BLOCK: First block number to write
SR176 : 0 to 15 ($00 to $0F)
SR512 family : 0 to 15 ($00 to $0F) or 255 ($FF)
SR4K family : 0 to 127 ($00 to $7F) or 255 ($FF)
NB: Number of blocks to write
SR176 : 1 to 16 ($01 to $10)
SR512 family : 1 to 16 ($01 to $10)
SR4K family : 1 to 60 ($01 to $3C)
DATA: Data to write, NB bytes, multiple of block size.

Note : SR176 has 2-byte blocks, SR512 and SR4K families have 4-byte blocks.

CLASS	INS	DATA OUT		
$06	$33	Length	Status	DATA (x)

Length: response length
Status: $00: communication interrupted
$01: bad CRC
$02: success
$03: bad parameters
$8x: Security activated (i.e. data written =/= data read)
$82: Security activated + good CRC
DATA: data read back (Length -1 byte)

$06 $34

Devices: *all printer models with ASK module*

Name: RELEASE (SR Family)
Description: Deactivation.

CLASS	INS	DATA IN
$06	$34	Param

Param: 00 : deactivation of the ticket using the “completion” instruction.
Others : RFU

CLASS	INS	DATA OUT
$06	$34	Status

Status: $00: ticket always active
$02: ticket deactivated

$06 $35

Devices: *all printer models with ASK module*

Name: READ (SR Family)
Description: Read bytes at a given address. This function handles the chip regardless the blocks organization.

CLASS	INS	DATA IN	
$06	$35	ADD (2)	NB

ADD: Address of the first reading : 2 bytes LSB, MSB
SR176 : 0 to 31 ($00 00 to $1F 00)
SR512 family : 0 to 63 ($00 00 to $3F 00) or 1020 to 1023 ($FC 03 to $FF 03)
SR4K family : 0 to 511 ($00 00 to $FF 01) or 1020 to 1023 ($FC 03 to $FF 03)
Note : ADD must be multiple of block size.

NB: Number of bytes to read
SR176 : 1 to 32 ($01 to $20)
SR512 family : 1 to 64 ($01 to $40)
SR4K family: 1 to 240 ($01 to $F0)

CLASS	INS	DATA OUT		
$06	$35	Length	Status	DATA (x)

Length: response length
Status: $00: communication interrupted
$01: bad CRC
$02: success
$03: bad parameters
DATA: data read (Length -1 byte)

$06 $36

Devices: *all printer models with ASK module*

Name: WRITE (SR Family)
Description: Write and verify bytes at a given address. This function handles the chip regardless the blocks organization.

CLASS	INS	DATA IN		
$06	**$36**	**ADD (2)**	**NB**	**DATA (x)**

ADD: Address of the first writing : 2 bytes LSB, MSB
SR176 : 0 to 31 ($00 00 to $1F 00)
SR512 family : 0 to 63 ($00 00 to $3F 00) or 1020 to 1023 ($FC 03 to $FF 03)
SR4K family : 0 to 511 ($00 00 to $FF 01) or 1020 to 1023 ($FC 03 to $FF 03)
Note : ADD must be multiple of block size.

NB: Number of bytes to write
SR176 : 1 to 32 ($01 to $20)
SR512 family : 1 to 64 ($01 to $40)
SR4K family : 1 to 240 ($01 to $F0)
Note : NB must be multiple of block size.

DATA: Data to write, NB bytes

CLASS	INS	DATA OUT		
$06	**$36**	**Length**	**Status**	**DATA (x)**

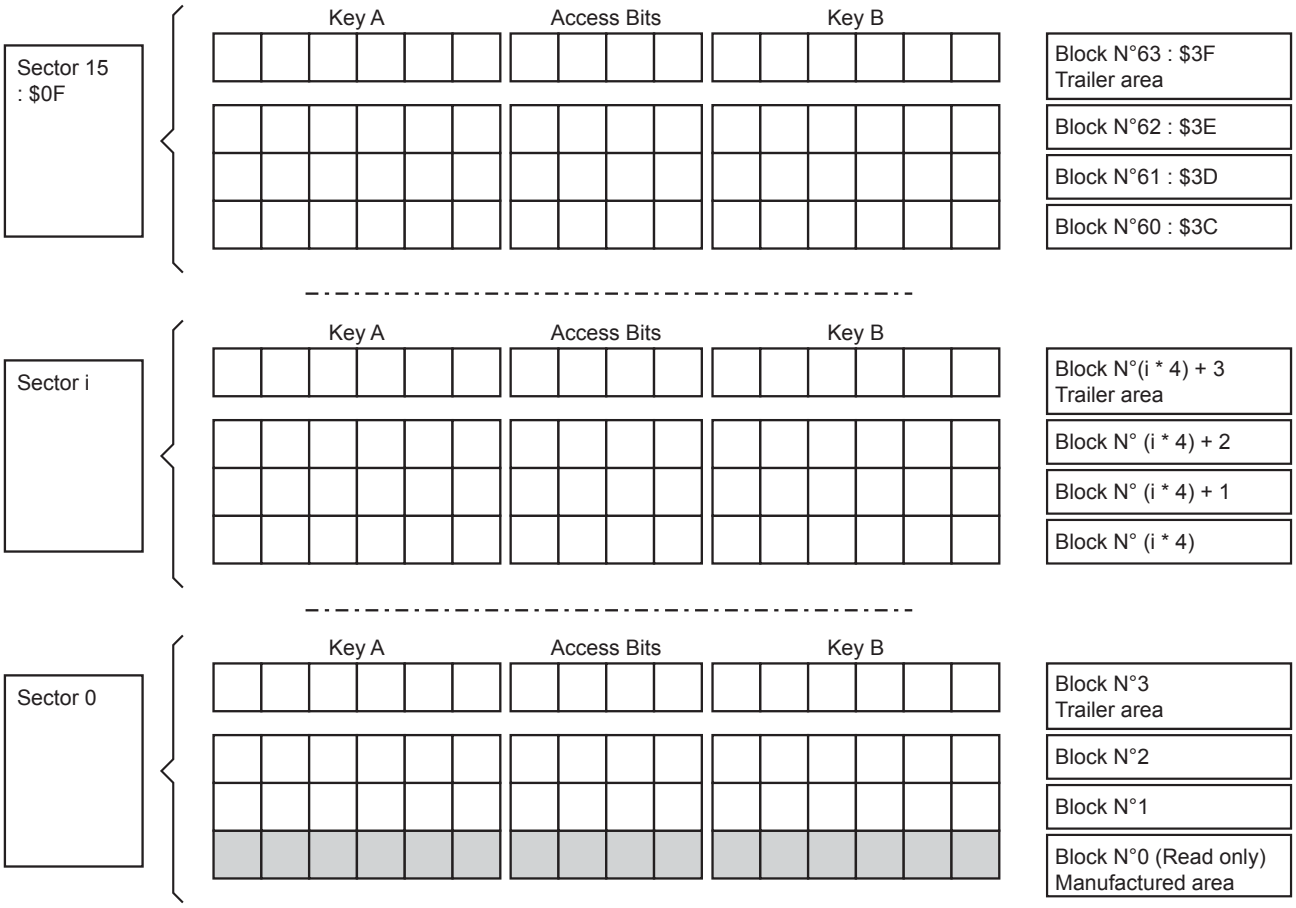
Length: response length
Status:
$00: communication interrupted
$01: bad CRC
$02: success
$03: bad parameters
$8x: Security activated (i.e. data written =/= data read)
$82: Security activated + good CRC
DATA: data read back (Length -1 byte)

3.7.10 MIFARE® Class (N°= $10)

The MIFARE® Class is based on the 14443 type A contactless protocol, it allows communicating and managing the MIFARE® Classic and MIFARE® 4K cards. Once in communication with a MIFARE® card it is not possible to communicate in the same time with a Type B card.
In addition, this class allow the management of the RCXXX chip and RF baurates.

3.7.10.1 MIFARE® Classis Cards (memory organization)

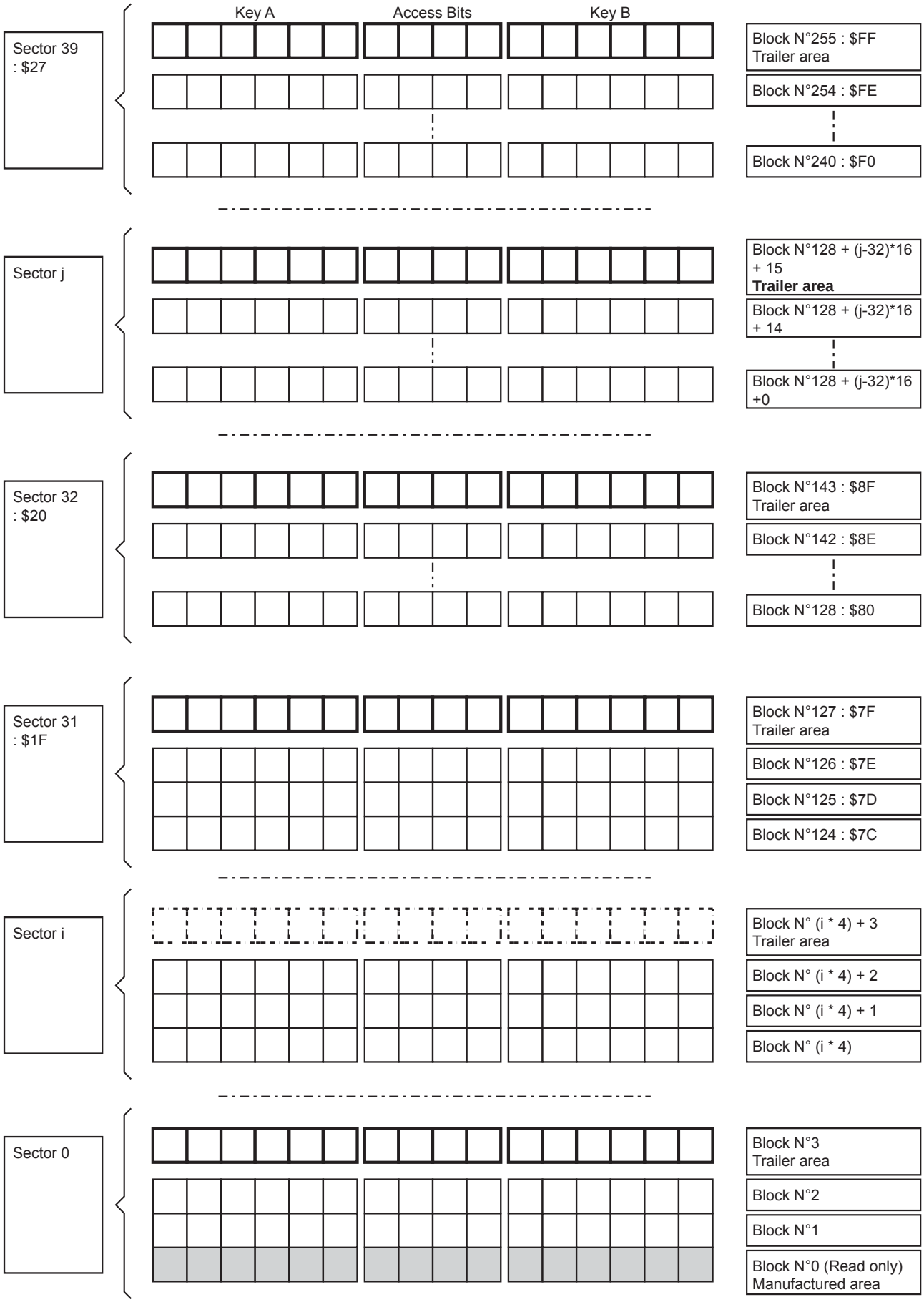
The MIFARE® Classic Card is composed of 16 Sectors of 4 blocks each. Each Block is composed of 16 Bytes.
The First Block of the First Sector is a read only block that contains the manufacturer information.
The Fourth Block of each sector contains a Trailer area, which contains the key and access bits for this sector.


Key A
Access Bits
Key B
Sector 15 : $0F
Block N°63 : $3F Trailer area
Block N°62 : $3E
Block N°61 : $3D
Block N°60 : $3C
Sector i
Block N°(i * 4) + 3 Trailer area
Block N° (i * 4) + 2
Block N° (i * 4) + 1
Block N° (i * 4)
Sector 0
Block N°3 Trailer area
Block N°2
Block N°1
Block N°0 (Read only) Manufactured area

For more details refer to the **MIFARE® STANDARD** user manual.

3.7.10.2 MIFARE® 4K Cards (memory organization)

The MIFARE® 4K Card is organised in 32 sectors with 4 blocks and 8 sectors with 16 blocks. Each Block is composed of 16 Bytes.
The First Block of the First Sector is a read only block that contains the manufacturer information.
The Fourth Block of the first 32 sectors and the sixteenth of the last 8 sectors contains a Trailer area, which contains the key and access bits for this sector.


Key A
Access Bits
Key B
Sector 39 : $27
Block N°255 : $FF Trailer area
Block N°254 : $FE
Block N°240 : $F0
Sector j
Block N°128 + (j-32)*16 + 15 Trailer area
Block N°128 + (j-32)*16 + 14
Block N°128 + (j-32)*16 +0
Sector 32 : $20
Block N°143 : $8F Trailer area
Block N°142 : $8E
Block N°128 : $80
Sector 31 : $1F
Block N°127 : $7F Trailer area
Block N°126 : $7E
Block N°125 : $7D
Block N°124 : $7C
Sector i
Block N° (i * 4) + 3 Trailer area
Block N° (i * 4) + 2
Block N° (i * 4) + 1
Block N° (i * 4)
Sector 0
Block N°3 Trailer area
Block N°2
Block N°1
Block N°0 (Read only) Manufactured area

For more details refer to the **MIFARE® 4K** user manual.

3.7.10.3 Remarks

- The commands Authenticate, ReadSector, ChangeKey authenticates a sector and load the PCD key specified in the internal memory of the RF chip.
- The commands ReadBlock, ReadMultipleBlock, WriteBlock, IncrementValue, DecrementValue and BackupRestoreValue need to be preceded by one of the previous command in order to authenticate the card sector.
- The cards can be detected in the field either by the command EnterHuntPhase or by the command DetectMF.
- For anticollision detection, the command DetectMF indicates the presence of more than one card by its status (0x18, just as the EnterHuntPhase in case of mono-card search) . In multicard mode, to select one specific card among all the cards present, the command selectMF must be used otherwise only the last detected card will be selected.

3.7.10.4 Access bit management

The Access Bits are coded redundantly on the bytes 6, 7 and 8 of the trailer block one of the 2 times in complement:

C1	C2	C3	Valid commands	Description
C1/3	C2/3	C3/3	Read/Write	Sector Trailer
C1/2	C2/2	C3/2	Read/Write, Increment/Decrement, Transfert/Restore	Data Block
C1/1	C2/1	C3/1	Read/Write, Increment/Decrement, Transfert/Restore	Data Block
C1/0	C2/0	C3/0	Read/Write, Increment/Decrement, Transfert/Restore	Data Block

Access Bits

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Byte 6	C2/3	C2/2	C2/1	C2/0	C1/3	C1/2	C1/1	C1/0
Byte 7	C1/3	C1/2	C1/1	C1/0	C3/3	C3/2	C3/1	C3/0
Byte 89	C3/3	C3/2	C3/1	C3/0	C2/3	C2/2	C2/1	C2/0

A bad coding of the access block "kills" the block access definitively.
The coding has a different meaning depending if the block is a trailer block or not.

Operation on Data Blocks

C1	C2	C3	Read	Write	Increment	Decrement Transfert Restore	Application
0	0	0	Key A | B	Key A | B	Key A | B	Key A | B	Transport Configuration
0	1	0	Key A | B	Never	Never	Never	Read/Write Block
1	0	0	Key A | B	Key B	Never	Never	Read/Write Block
1	1	0	Key A | B	Key B	Key B	Key A | B	ValueBlock
0	0	1	Key A | B	Never	Never	Key A | B	ValueBlock
0	1	1	Key B	Key B	Never	Never	Read/Write Block
1	0	1	Key B	Never	Never	Never	Read/Write Block
1	1	1	Never	Never	Never	Never	Read/Write Block

Key and access Operation on trailer block

C1	C2	C3	Key A		Access Bits		Key B		Remark
			Read	Write	Read	Write	Read	Write	
0	0	0	Never	Key A	Key A	Never	Key A	Key A	KeyB can't authenticate
0	1	0	Never	Never	Key A	Never	Key A	Never	KeyB can't authenticate
1	0	0	Never	Key B	Key A | B	Never	Never	Key B	
1	1	0	Never	Never	Key A | B	Never	Never	Never	
0	0	1	Never	Key A	Key A	Key A	Key A	Key A	KeyB can't authenticate
0	1	1	Never	Key B	Key A | B	Key B	Never	Key B	
1	0	1	Never	Never	Key A | B	Key B	Never	Never	
1	1	1	Never	Never	Key A | B	Never	Never	Never	

3.7.10.5 List of Error Codes

Decimal Value	Hexadecimal Value	Code	Description
0	$00	MF_OK	Normal Execution
1	$01	MF_NOTAGERR	No card response (Timout or card removed)
2	$02	*MF_CRCERR*	Reserve for future use
3	$03	*MF_EMPTY*	Reserve for future use
4	$04	MF_AUTHERR	Authentication Error (GetKey or Auth command)
5	$05	*MF_PARITYERR*	Reserve for future use
6	$06	MF_CODEERR	Coding Error detected during RF reception
7	$07	MF_NONXPCHIP	No Nxp Chip family detected
8	$08	*MF_SERNRERR*	Reserve for future use
9	$09	*MF_EEPROMERR*	Reserve for future use
10	$0A	MF_NOTAUTHERR	Command was passed on unauthenticated sector
11	$0B	MF_BITCOUNTERR	Error occurs on number of bits of card response
12	$0C	MF_BYTECOUNTERR	Error occurs on number of bytes of response
13	$0D	*MF_IDLE*	Reserve for future use
14	$0E	*MF_TRANSERR*	Reserve for future use
15	$0F	MF_WRITEERR	Write Error
16	$10	*MF_VALERR*	Reserve for future use
17	$11	MF_KEYERR	Key Error detected (GetKey command)
18	$12	*MF_READERR*	Reserve for future use
19	$13	MF_OVFLERR	Overflow Error detected (any command)
20	$14	*MF_POLLING*	Reserve for future use
21	$15	MF_FRAMINGERR	Framing Error detected (any command)
22	$16	*MF_ACCESSERR*	Reserve for future use
23	$17	MF_UNKNOWN_COMMAND	Unknown Command detected (any command)
24	$18	MF_COLLERR	Collision Error detected (any command)
25	$19	*MF_RESETERR*	Reserve for future use
26	$1A	*MF_INTERFACEERR*	Reserve for future use
27	$1B	MF_ACCESSTIMEOUT	Access Timeout on any commands.
28	$1C	MF_BCCERR BCC	Error (Select command) bad Check code
29	$1D	MF_SAKERR SAK	Error (Select command) not acknowledged
30	$1E	MF_PICCKEYIDEER	Picc Key Index Error (Authenticate command)
31	$1F	MF_PICCACCESSBITERR	Incoherent ACCESS BIT condition
32	$20	MF_ERRBV	Error on check block value after value operation
60	$3C	MF_WRONGPARAM	Wrong parameters in the command.
100	$64	MF_NYIMPLEMENTED	No Key Implemented or other error.
120	$78	*MF_COMM_ABORT*	Reserve for future use
121	$79	*MF_CALLOPEN*	Reserve for future use

In Italic are the Error codes not implemented yet.

3.7.10.6 Reader function

$10 $01

Devices: *all printer models with ASK module*

Name: SENDNXP (MIFARE®)
Description: Manage the Mifare® Chip.
Theses functions are available but shouldn't be used without a very good knowledge of the Nxp chip family.

A) Maintenance Functions:
Details: Indicate the command performed on the RF head chip at Initialization level

CLASS	INS	DATA IN	
$10	$01	Length	TYPE

Length: 1 byte: $01 for the following commands
TYPE: 1 byte:
- $00 Initialize the chip (performed automatically at CSC reset
- $01 Disable chips (Needed only to turn power down on chip.
- $02 Load of the default configuration in EEPROM. Should only be used at manufacturing level.
- $03 Reset the Chip (The default ASK configuration is set)

Returned response

CLASS	INS	DATA OUT	
$10	$01	Length	Status

Length: 1 byte: $01 for this command
Status: 1 byte:
$00: Chip detected and correct operation
$07: Chip not detected
$09: Chip EEPROM Error

B) Get Key from storage area to Internal RAM Buffer:
on GEN5XX keys are not remaining after a reset or power down of the coupler.

CLASS	INS	DATA IN		
$10	$01	Length	TYPE	KEY Index

Length: 1 byte: $02 for this command
TYPE: 1 byte: $04 type for the command that get Key from storage area: Load Key KEYIndex from storage area by its index in the internal buffer for the next cryptographic operation.
KEYIndex: 1 byte: value from $00 to $1F index of one of the 32 keys

Returned response

CLASS	INS	DATA OUT	
$10	$01	Length	Status

Length: 1 byte: $01 for this response
Status: 1 byte: idem previous Status

C) Dump Current Configuration:

CLASS	INS	DATA IN	
$10	**$01**	**Length**	**Type**

Length: 1 byte: $01 for this command
Type: 1 byte: $05 type of the command that Dump Current configuration of all the internal register of the chip. This function is only useful for expertise of problem relating to the Chip.

CLASS	INS	DATA OUT		
$10	**$01**	**Length**	**Status**	**32REGContent (32 bytes)**

Length: 1 byte: $21 in case of success for this Response
Status: 1 byte: idem previous Status:
32REGContent: 32 bytes: The content of the 32 first registers is returned

D) Get fake Serial Number:

CLASS	INS	DATA IN	
$10	**$01**	**Length**	**Type**

Length: 1 byte: $01 for this command
Type: 1 byte: $09 type of the command that gives the fake Serial Number of the chip for compatibility.

Returned response

CLASS	INS	DATA OUT		
$10	**$01**	**Length**	**Status**	**FakeSerialNumber (4 bytes)**

Length: 1 byte: $05 in case of success for this Response
Status: 1 byte: idem previous Status:
FakeSerialNumber: 4 bytes: for GEN5XX it should be a fixed value : 0x80, 0x80, 0x80, 0x80

E) Load Internal Key Set:
This function can be used before an operation on a new sector.
Be careful: The Byte order for the KeyValue is different than the one used in the ChangeKey Command.
N.B. : Keys need to be reloaded after each powerup.

CLASS	INS	DATA IN			
$10	**$01**	**Length**	**Type**	**PCDIndex (1 byte)**	**Key value (6 bytes)**

Length: 1 byte: $08 for this command
Type: 1 byte: $06 type of the command to Load an internal Key in the chip.
PCDIndex : 1 byte: the PCDIndex parameter contains the Key Index from 0x00 to 0x1F
Key value: 6 bytes: the Key is transmitted unencrypted on 6 bytes MSB First

Returned response

CLASS	INS	DATA OUT	
$10	**$01**	**Length**	**Status**

Length: 1 byte: $01 for this Response
Status: 1 byte: idem previous Status

F) Load Key into the Internal Buffer:
This function can be used before any operation which needs cryptographic work to specify the key to use.

CLASS	INS	DATA IN		
$10	$01	Length	TYPE	Key value (6 bytes)

Length: 1 byte: $07 for this command
Type: 1 byte: $0B type of the command to Load a Key into the internal Buffer.
Key value: 6 bytes: the Key is transmitted unencrypted on 6 bytes MSB First

Returned response

CLASS	INS	DATA OUT	
$10	$01	Length	Status

Length: 1 byte: $01 for this Response
Status: 1 byte: idem previous Status

G) SET Chip Proper Configuration:

CLASS	INS	DATA IN		
$10	$01	Length	Type	Mode

Length: 1 byte: $02 for this command
Type: 1 byte: $0C type of the command to set a proper configuration into the Chip.
Mode: 1 byte: 00 => ISO14443 B
01 => ISO14443 A
02 => FELICA

Returned response

CLASS	INS	DATA OUT	
$10	$01	Length	Status

Length: 1 byte: $01 for this Response
Status: 1 byte: idem previous Status

H) Get Chip identification

CLASS	INS	DATA IN	
$10	$01	Length	Type

Length: 1 byte: $01 for this command
Type: 1 byte: $0D type of the command to know information about the chip.

Returned response

CLASS	INS	DATA OUT			
$10	$01	Length	Status	ID chip (4 bytes)	Chip Version (1 byte)

Length: 1 byte: $06 for this Response
Status: 1 byte: idem previous Status
ID chip: 4 bytes (on GEN5XX it should be the fixed value 0x80, 0x80, 0x80, 0x80)
Chip version: 1 byte (on GEN5XX it should be the fixed value 0x80)

I) WRITE Internal Register:

CLASS	INS	DATA IN			
$10	**$01**	**Length**	**Type**	**Register Address (1 byte)**	**Value (1 byte)**

Length: 1 byte: $03 for this command
Type: 1 byte: $0E type of the command to write new value at the specified register address.
Register Address: 1 byte: specified register address
Value: 1 byte: new value to write

Returned response

CLASS	INS	DATA OUT	
$10	**$01**	**Length**	**Status**

Length: 1 byte: $01 for this Response
Status: 1 byte: idem previous Status + $0F: Chip WRITE Error

J) READ Internal Register:

CLASS	INS	DATA IN		
$10	**$01**	**Length**	**Type**	**Register Address (1 byte)**

Length: 1 byte: $02 for this command
Type: 1 byte: $0F type of the command to read value at the specified register address.
Register Address: 1 byte: specified register address

Returned response

CLASS	INS	DATA OUT		
$10	**$01**	**Length**	**Status**	**Value**

Length: 1 byte: $02 for this Response
Status: 1 byte: idem previous Status + $12: Chip READ Error
Value: 1 byte: data read in the register

K) Load Speed RF limitation:

CLASS	INS	DATA IN				
$10	**$01**	**Length**	**Type**	**Rx**	**Tx**	**Don't negociate**

Length: 1 byte: $04 for this command
Type: 1 byte: $10 type of the command to set RF speed limitation.
Rx: 1 byte: $00 => 106 kb/ s
$01 => 212 kb/ s
$02 => 424 kb/ s
$03 => 824 kb/ s
Tx: 1 byte: $00 => 106 kb/ s
$01 => 212 kb/ s
$02 => 424 kb/ s
$03 => 824 kb/ s
Don’t negotiate: 1 byte: $00 False
$01 True

Returned response

CLASS	INS	DATA OUT	
$10	**$01**	**Length**	**Status**

Length: 1 byte: $01 for this Response
Status: 1 byte: idem previous Status

L) Write RF chip register in EEPROM:

CLASS	INS	DATA IN			
$10	$01	Length	Type	Address	Data

Length: 1 byte: $03 for this command
Type: 1 byte: $12 type of the command to write in eeprom.
Address: 1 byte: address of register
Data: 1 byte: data value

Returned response

CLASS	INS	DATA OUT	
$10	$01	Length	Status

Length: 1 byte: $01 for this Response
Status: 1 byte: idem previous Status

M) Get Status:

CLASS	INS	DATA IN	
$10	$01	Length	Type

Length: 1 byte: $01 for this command
Type: 1 byte: $13 type of the command to get chip status.

Returned response

CLASS	INS	DATA OUT	
$10	$01	Length	Status

Length: 1 byte: $06 for this Response
Status: 6 bytes: RegPrimaryStatus, RegSecondaryStatus, RegErrorFlag, Status, ISO4 Rx Retries Level, ISO4 Rx Retries

N) Set custom Frame Waiting Time

CLASS	INS	DATA IN			
$10	$01	Length	Type	Hi	Low

Length: 1 byte: $03 for this command
Type: 1 byte: $14 type of the command to set custom FWT
Hi: 1 byte: high byte of custom FWT.
Low: 1 byte: low byte of custom FWT.

FWT = Hi*256 + Low. If FWT = 0, FWT is given by the card during detection.

Returned response

CLASS	INS	DATA OUT	
$10	$01	Length	Status

Length: 1 byte: $01 for this Response
Status: 1 byte: idem previous Status

O) Get current RF speed (DRI and DSI)

CLASS	INS	DATA IN	
$10	**$01**	**Length**	**Type**

Length: 1 byte: $01 for this command
Type: 1 byte: $15 type of the command to get current DRI and DSI

Returned response

CLASS	INS	DATA OUT			
$10	**$01**	**Length**	**Status**	**DSI**	**DRI**

Length: 1 byte: $02 for this Response
Status: 1 byte: idem previous Status
DSI: 1 byte: PICC to PCD RF speed
DRI: 1 byte: PCD to PICC RF speed

DSI and DRI coding : 00 → 106 kb/s
01 → 212 kb/s
02 → 424 kb/s
03 → 847 kb/s

P) Get current Frame Waiting Time

CLASS	INS	DATA IN	
$10	**$01**	**Length**	**Type**

Length: 1 byte: $01 for this command
Type: 1 byte: $16 type of the command to get current FWT

Returned response

CLASS	INS	DATA OUT		
$10	**$01**	**Length**	**Hi**	**Low**

Length: 1 byte: $02 for this Response
Hi: 1 byte: high byte of current FWT.
Low: 1 byte: low byte of current FWT.

FWT = Hi*256 + Low.

3.7.10.7 Card functions

$10 $02

Devices: *all printer models with ASK module*

Name: Select MF (MIFARE®)
Description: This command selects one specific MIFARE® card present in the field by its “Unique ID”. It allows in case of collision to select one card. This command requires the RF chip. This function realizes the ISO14443 connection such as REQA and SELECT.

CLASS	INS	DATA IN
$10	$02	Picc serial Number (4 bytes)

Picc Serial Number: 4 bytes: Serial number of the card

CLASS	INS	DATA OUT			
$10	$01	Length	Status	Code	Picc serial Number (4 bytes)

Length: 1 byte: Response length: $06 in case of success for this Response
Status: 1 byte: See List of response code
Code: 1 byte: $08 means MIFARE® STANDARD 1K Card, $18 MIFARE® 4K, 0x28 for ProX.
Picc Serial Number: 4 bytes: Serial number of the card

$10 $03

Devices: *all printer models with ASK module*

Name: Change Key (MIFARE®)
Description: This function allows writing in the trailer block of a PICC sector to change the keys and dedicated access rights. In a first step the sector is authenticated with the old key. In a second step the trailer block is modified with this same key and then re-authenticated with this same key or the new parameters.
Remark: The keys are in the inverse order than the LoadKey Command
Caution: The use of this function supposes to know the key present in the PCD and in the PICC and a perfect knowledge of the access bits condition on the MIFARE® Picc. The change of key operation is not always possible depending on the previous choice of the access bit

CLASS	INS	DATA IN								
$10	$03	Length	KeyPICC	SectorNum	KeyPCD	NewKeyA	AccessBits	FreeByte	NewKeyB	FinalAutKey

Length: 1 byte: $14 for this command
KeyPICC: 1 byte: value $0A for KEYA or $0B for KEYB used for initial authentication (default is A)
SectorNum: 1 byte: Sector Number of the sector (value from $00 to $0F for a MIFARE® STANDARD 1K Card, and from $00 to $27 for a MIFARE® 4K) on which the keys have to be changed
KeyPCD: 1 byte: Index of the PCD key (value $00 to $1F for storage area keys, $FF:internal buffer Key) used for initial authentication
NewKeyA: 6 bytes: New value of the A Key (unencrypted) with LSB First
AccessBits: 3 bytes: All access bits including trailer block as described in the MIFARE® specification.
FreeByte : 1 byte: Last byte of the access bit of the trailer block, its value can be used for any purpose.
NewKeyB: 6 bytes: New value of the B Key (unencrypted) with LSB First
FinalAutKey: 1 byte: Index of the key value for final Authentication $0A for KEYA or $0B for KEYB

CLASS	INS	DATA OUT			
$10	$03	Length	Status	Code	Picc serial Number (4 bytes)

Length: 1 byte: Response length: $06 in case of success for this Response
Status: 1 byte: See List of response code
Code: 1 byte: $08 means MIFARE® STANDARD 1K Card, and $18 MIFARE® 4K.
Picc Serial Number: 4 bytes: Serial number of the card

$10 $04

Devices: *all printer models with ASK module*

Name: Detect MF (MIFARE®)
Description: This function allows to detect the MIFARE® card present in the antenna field and select it (In case of collision a specific error code is returned). No key is required at this stage in the RF chip. This function realizes the ISO14443 connection Norm operations such as REQA and SELECT. There is no parameter needed.

CLASS	INS	DATA IN
$10	$04	-

No input data

CLASS	INS	DATA OUT			
$10	$04	Length	Status	Code	Picc serial Number (4 bytes)

Length: 1 byte: Response length: $06 in case of success for this Response
Status: 1 byte: See List of response code (0x18 for collision between two cards of the same kind (1K or 4K) and 0x1B for collision between one MIFARE® STANDARD 1K and a MIFARE® 4K).
Code: 1 byte: $08 means MIFARE® STANDARD 1K Card, $18 MIFARE® 4K and 0x28 for ProX
Picc Serial Number: 4 bytes: Serial number of the card

$10 $05

Devices: *all printer models with ASK module*

Name: Authenticate MF (MIFARE®)
Description: This function authenticates a sector. The right key (according to the block to authenticate: operation can be done only for one sector at a time) is required in the internal buffer of the RF chip. The Sector number to authenticate and indexes of key in the PCD and the PICC are transmitted as parameters in DataIn.

CLASS	INS	DATA IN			
$10	$05	Length $03	KeyPICC	SectorNum	KeyPCD

Length: 1 byte: fixed value = $03
KeyPICC: 1 byte: value $0A for KEYA or $0B for KEYB
SectorNum: 1 byte: Sector Number of the sector to authenticate value from $00 to $0F for a MIFARE® STANDARD 1K Card, and from $00 to $27 for a MIFARE® 4K.
KeyPCD : 1 byte: Index of the PCD key, value $00 to $1F for storage area keys, $FF:internal buffer Key.

CLASS	INS	DATA OUT			
$10	$05	Length	Status	Code	Picc serial Number (4 bytes)

Length: 1 byte: Response length
Status: 1 byte: See List of response code
Code: 1 byte: $08 means MIFARE® STANDARD 1K Card, and $18 MIFARE® 4K.
Picc Serial Number: 4 bytes: Serial number of the card

$10 $06

Devices: *all printer models with ASK module*

Name: ReadBlock (MIFARE®)
Description: The aim of this function is to read an authenticated block, the right key according with the one of the block to authenticate must be present in the internal buffer of the RF chip.

CLASS	INS	DATA IN	
$10	$06	Length	BlockNum

Length: 1 byte: fixed value = $01
BlockNum : 1 byte: authenticated Block Number to read value from $00 to $3F for a MIFARE® STANDARD 1K Card, and from $00 to $FF for a MIFARE® 4K.

CLASS	INS	DATA OUT		
$10	$06	Length	Status	BlockContent (16 bytes)

Length: 1 byte: Response length
Status: 1 byte: See List of response code
BlockContent: 16 bytes: Block content read in the card
Remark: This command sends back the Type and the Serial Number of the card
In case of bad Transmission Error (0x0C,../..)

$10 $07

Devices: *all printer models with ASK module*

Name: ReadSector (MIFARE®)
Description: The aim of this function is to read a Sector of the PICC.

CLASS	INS	DATA IN			
$10	$07	Length	KeyPICC	SectorNum	KeyPCD

Length: 1 byte: fixed value = $03
KeyPICC: 1 byte: value $0A for KEYA or $0B for KEYB
SectorNum: 1 byte: Sector Number of the sector to authenticate value $00 to $0F
KeyPCD : 1 byte : Index of the PCD key value $00 to $1F for storage area keys, $FF:internal buffer Key.

CLASS	INS	DATA OUT				
$10	$07	Length	Status	Code	PiccSerialnumber (4 bytes)	SectorContent (N)

Length: 1 byte : Response length $46 in case of successful operation
Status: 1 byte : See List of response code
Code: 1 byte: $08 means MIFARE® STANDARD 1K Card, and $18 MIFARE® 4K.
Picc Serial Number: 4 bytes: Serial number of the card
SectorContent: 64 bytes for a MIFARE® STANDARD 1K Card and 256 for a MIFARE® 4K: Sector read in the card in the following order : [Block0] [Block1] [Block2] … The [Block3] (or [Block16] for a MIFARE® 4K)is the Trailer Block which includes the Keys and access bit, the value read may not be the right one depending on the access bits.

NB: For a MIFARE® 4K Card, if SectorNum >= 0x28, the read function will loop back at the beginning of the sectors of the card. It is identical for the MIFARE® STANDARD 1K except that the Sectors from 0x10 to 0x27 are prohibited.
NB: For a MIFARE® 4K Card, if 256 bytes have to be returned, the returned Length will appear on 2 bytes, with the same rule as the CSC frames (see chapter 3.2.6 Frame length)

$10 $08

Devices: *all printer models with ASK module*

Name: WriteBlock (MIFARE®)
Description: This function writes an authenticated block. The right key according with the block to write needs to be present in the internal buffer of the RF Chip.
Caution : A block will always be written entirely (16 bytes) This command can be used for data block as well as for block trailer but in this case access bit would not be verified (the ChangeKey command is safer for block trailer operation).
Remark : Due to card problem, the Write Block on block Trailer 3 sector 0 fails 1 over 2 times on the re-read command.

CLASS	INS	DATA IN		
$10	$08	Length	BlockNum	DataToWrite

Length : 1 byte : length ($11)
BlockNum : 1 byte : authenticated Block Number to be written.
DataToWrite : 16 bytes : Data to write in the selected authenticated block

CLASS	INS	DATA OUT		
$10	$08	Length	Status	DataVerification

Length: 1 byte : Response length $11 for successful operation
Status: 1 byte : See List of response code
DataVerification: 16 bytes: content of the memory read after operation

$10 $09

Devices: *all printer models with ASK module*

Name: Halt MF (MIFARE®)
Description: The aim of this function is to send the HALT order on active PICC.

CLASS	INS	DATA IN
$10	$09	-

No parameters required

CLASS	INS	DATA OUT	
$10	$08	Length	Status

Length: 1 byte: Response length
Status: 1 byte: See List of response code

$10 $0D

Devices: *all printer models with ASK module*

Name: ReadMultipleBlock (MIFARE®)
Description: The aim of this function is to read several blocks in an authenticated sector, the right key according with the one of the block to authenticate must be present in the internal buffer of the RF Chip.

CLASS	INS	DATA IN		
$10	**$0D**	**Length**	**BlockNum**	**Number**

Length: 1 byte: fixed value = $02
BlockNum : 1 byte: authenticated Block Number to read value from $00 to $3F for a MIFARE® STANDARD 1K Card, and from $00 to $FF for a MIFARE® 4K.
Number: 1 byte: Number of blocks to read, in a single sector. Otherwise an authentication error will be sent back.

CLASS	INS	DATA OUT		
$10	**$08**	**Length**	**Status**	**BlockContent (16 bytes)**

Length: 1 byte: Response length
Status: 1 byte: See List of response code
BlockContent: Blocks content read in the card
Remark: This command sends back the Type and the Serial Number of the card in case of bad Tran smission Error(0x0C,../..)

$10 $0E

Devices: *all printer models with ASK module*

Name: SimpleWriteBlock (MIFARE®)

Description: This function writes an authenticated block. The right key according with the block to write needs to be present in the internal buffer of the RF Chip. SELECT must have been realised, otherwise write will fail.
Caution : A block will always be written entirely (16 bytes) This command can be used for data block as well as for block trailer but in this case access bit would not be verified (the ChangeKey command is safer for block trailer operation).
Remark : Due to card problem, the Write Block on block Trailer 3 sector 0 fails 1 over 2 times on the re-read command.

CLASS	INS	DATA IN		
$10	$0E	Length	BlockNum	DataToWrite

Length : 1 byte : length ($11)
BlockNum : 1 byte : authenticated Block Number to be written.
DataToWrite : 16 bytes : Data to write in the selected authenticated block

CLASS	INS	DATA OUT	
$10	$0E	Length	Status

Length: 1 byte : Response length $01 for successful operation
Status: 1 byte : See List of response code

$10 $0F

Devices: *all printer models with ASK module*

Name: ReadSectorData (MIFARE®)
Description: The aim of this function is to read the data blocks of a Sector of the PICC.

CLASS	INS	DATA IN			
$10	$0F	Length	KeyPICC	SectorNum	KeyPCD

Length: 1 byte: fixed value = $03
KeyPICC: 1 byte: value $0A for KEYA or $0B for KEYB
SectorNum: 1 byte: Sector Number of the sector to authenticate value $00 to $0F
KeyPCD : 1 byte : Index of the PCD key value $00 to $1F for storage area keys, $FF:internal buffer Key.

CLASS	INS	DATA OUT				
$10	$0F	Length	Status	Code	PiccSerialNumber (4 bytes)	SectorContent (N)

Length: 1 byte : Response length $36 in case of successful operation
Status: 1 byte : See List of response code
Code: 1 byte: $08 means MIFARE® STANDARD 1K Card, and $18 MIFARE® 4K.
Picc Serial Number: 4 bytes: Serial number of the card
SectorContent: 48 bytes for a MIFARE® STANDARD 1K Card or 240 bytes for a MIFARE® 4K:
Sector read in the card in the following order : [Block0] [Block1] …[Block2] (or [Block15] for a MIFARE® 4K).

NB: For a MIFARE® 4K Card, if SectorNum >= 0x28, the read function will loop back at the beginning of the sectors of the card. It is identical for the MIFARE® STANDARD 1K except that the Sectors from 0x10 to 0x27 are prohibited.

$10 $10

Devices: *all printer models with ASK module*

Name: WriteSectorData (MIFARE®)
Description: The aim of this function is to write the data blocks of a Sector of the PICC.

CLASS	INS	DATA IN				
$10	$10	Length	KeyPICC	SectorNum	KeyPCD	DataToWrite

Length: 1 byte: fixed value = $33 for Mifare1K or $F3 for Mifare4K
KeyPICC: 1 byte: value $0A for KEYA or $0B for KEYB
SectorNum: 1 byte: Sector Number of the sector to authenticate value $00 to $0F
KeyPCD : 1 byte : Index of the PCD key value $00 to $1F for storage area keys, $FF:internal buffer Key.
DataToWrite : 48 (or 240) bytes * : Data to write in the selected sector

CLASS	INS	DATA OUT	
$10	$10	Length	Status

Length: 1 byte : Response length $01 for successful operation
Status: 1 byte : See List of response code

3.7.10.8 Value block functions

N.B. : To initialize a block as a "value block" by the access bits, first the data block must be initialized by a WriteBlock command as follow :

Byte Number	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Description	Value				Value				Value				Ad	Ad	Ad	Ad

On an increment or a decrement command, the PICC does a hardware update of the 3 fields containing value and complement of this value. The Address of the block (Ad bytes) must be present but they are not updated in the operation.

$10 $0A

Devices: *all printer models with ASK module*

Name: Increment Value (MIFARE®)
Description: This function is used to increment a PICC block initialized as a "value block" which is a kind of counter (see chapter access control bits). The block need to be previously authenticated with the right key dedicated to the increment operation. The max increment value can stand on 4 bytes. (Overflow must be managed by the application).
N.B. The use of this function requires the knowledge of the Key and access bits present in the Picc and the Pcd.

CLASS	INS	DATA IN		
$10	**$0A**	**Length**	**BlockPicc-**	**Increment (B0,B1,B2,B3)**

Length: 1 byte : length fixed at $05 for this command
BlockPicc: 1 byte : block number from $00 to $3F
Increment: 4 bytes : value to increment on the counter (LSB first)

Returned response format

CLASS	INS	DATA OUT		
$10	**$0A**	**Length**	**Status**	**ControlValue (B0,B1,B2,B3)**

Length: 1 byte : Response length $05 for successful operation
Status: 1 byte : See List of response code
ControlValue : 4 bytes : Value after increment operation for control purpose (MSB first).

i.e. : 10 0A 05 04 01 00 00 00 (command for an increment of 1 on the value block 4)
gives for example the response 10 0A 05 00 00 00 01 02 if the previous value was 0x101 the new value is 0x102

$10 $0B

Devices: *all printer models with ASK module*

Name: Decrement Value (MIFARE®)
Description: This function is used to decrement a PICC block initialized as a "value block" which is a kind of counter (see chapter access control bits). The block need to be previously authenticated with the right key dedicated to the decrement operation. The max decrement value can stand on 4 bytes. (Underflow must be managed by the application).

N.B. The use of this function requires the knowledge of the Key and access bits present in the Picc and the Pcd.

CLASS	INS	DATA IN		
$10	**$0B**	**Length**	**BlockPicc-**	**Decrement (B0,B1,B2,B3)**

Length: 1 byte : length fixed at $05 for this command
BlockPicc: 1 byte : block number from $00 to $3F
Increment: 4 bytes : value to increment on the counter (LSB first)

Returned response format

CLASS	INS	DATA OUT		
$10	**$0B**	**Length**	**Status**	**ControlValue (B0,B1,B2,B3)**

Length: 1 byte : Response length $05 for successful operation
Status: 1 byte : See List of response code
ControlValue : 4 bytes : Value after increment operation for control purpose (MSB first).

i.e. : 10 0B 05 04 01 00 00 00 (command for an decrement of 1 on the value block 4)
Gives for example the response 10 0B 05 00 00 00 01 01 if the previous value was 0x102 the new value is 0x101

$10 $0C

Devices: *all printer models with ASK module*

Name: BackUp or Restore a Value Block (MIFARE®)

Description: This function can be use as well as a "Backup Command" or a "Restore Command". The aim of this command set is to restore the previous value of a block as anti-tearing protection (when the card is getting out of the field during a write operation). The previous value can be restored from another "value block" location.
To do so the two blocks must belong to the same sector of the PICC.
The block that need restoration (the one which will be written) need to be previously authenticated with the key dedicated to the "Restore" operation (see access bit condition in the Mifare® documentation)

N.B. The use of this function requires the knowledge of the Key and access bits present in the Picc and the Pcd.

CLASS	INS	DATA IN		
$10	**$0C**	**Length**	**BackupBlock-**	**RestoreBlock**

Length: 1 byte : length fixed at $02 for this command
BackupBlock: 1 byte : backup block number from $00 to $3F
(where the value was copied before the failed operation)
RestoreBlock: 1 byte : Restore block number from $00 to $3F
(where the value must be copied to restore its previous value)

Returned response format

CLASS	INS	DATA OUT		
$10	**$0C**	**Length**	**Status**	**ControlValue (B0,B1,B2,B3)**

Length: 1 byte : Response length $05 for successful operation
Status: 1 byte : See List of response code
ControlValue : 4 bytes : Value after restore operation for control purpose (MSB first).

N.B. : On a restore command, the PICC makes a hardware update of the 3 fields containing value and complement of value. The Address (Ad) bytes have nothing to do with the address managed by the restore command (considered as value during the restoration).

3.7.11 DESFIRE® Class (N°= $11)

The DESFIRE® Class is based on the NXP® DESFIRE card.
Only function accessible with a SAM MAC are available.
That Class of function is available only if activated by ASK.

3.7.11.1 DESFIRE® Cards Memory organization

Flexible file system
Up to 28 applications simultaneously on one PICC
Up to 16 files in each application

The 2 or 4 kbyte NV-memory is organised using a flexible file system. This file system allows a maximum of 28 different applications on one single PICC. Each application provides up to 16 files. Each application is represented by it's 3 bytes Application IDentifier, AID. Five different file types are supported

3.7.11.2 DESFIRE® Cards Access Commands

Application management:
- CreateApplication
- DeleteApplication
- SelectApplication
- FormatPICC
- GetApplicationIDs

Security management:
- Session
  - AbortTransaction
  - Authenticate
  - ComitTransaction
- Change Key
  - ChangeKey
  - ChangeKeySettings
  - GetKeySettings
  - GetKeyVersion
- GetVersion

File management:
- ChangeFileSetings
- ClearRecordFile
- CreateBackupDataFile
- CreateCyclicRecordFile
- CreateLinearRecordFile
- CreateStdDataFile
- CreateValueFile
- Credit
- Debit
- DeleteFile
- GetFileIDs
- GetFileSettings
- GetValue
- LimitedCredit
- ReadData
- ReadRecords
- WriteData
- WriteRecord

3.7.11.3 DESFIRE® Error List

DESFire Card Response Data are of type SW1=0x91 SW2=0xYY
And DesFire Sam response Data are of type SW1=0x90 SW2=0xYY

Sw2(hex)	Status	Description
0x00	Operation ok	Successful operation
0x0C	No changes	No changes done to backup files, CommitTransaction / AbortTransaction not necessary
0x0E	Out of eeprom error	Insufficient Non Volatile memory to complete command
0x1C	Illegal command code	Command code not supported
0x1E	Integrity error	CRC or MAC does not match data
0x40	No such key	Invalid key number specified
0x7E	Length error	Length of command string invalid
0x9D	Permission denied	Current configuration/Status does not allow the requested command
0x9E	Parameter Error	Value of the parameter is invalid
0xA0	Application not found	Request AID not present on PICC
0xA1	Appli. Integrity error	Unrecoverable error within application, application will be disabled
0xAE	Authentication error	Current authentication status does not allow the requested command
0xAF	Additional frame	Additional data frame is expected to be sent
0xBE	Boundary error	Attempt to read/write data from/to beyond the file's/record's limits.
0xC1	PICC integrity error	Unrecoverable error within PICC, PICC will be disabled
0xCA	Command aborted	Previous command was not fully completed (not all frame were requested or provided)
0xCD	PICC disabled error	Picc was disabled by an unrecoverable error.
0xCE	Count error	Number of applications limited to 28, no additional Create Application possible.
0xDE	Duplicate error	Creation of file/application failed because file/application with same number already exists.
0xEA	No DesFireSam	No DesFire Sam is available.
0xEE	EEprom error	Could not complete NV write operation due to loss of power, internal backup/rollback mechanism activated
0xF0	File not found	Specified file number does not exist
0xF1	File integrity error	Unrecoverable error within file, file will be disabled
0xFC	Bad Length	Bad SAM response
0xFD	Bad Param	Bad parameter Applicative level the command is not transmitted to the card
0xFE	Bad Length	Bad length parameter Appli level (buffer overload … command is not given to the card)
0xFF	Timeout	No response from the card

3.7.11.4 DESFIRE® set of instruction

$11 $01

Devices: *all printer models with ASK module*

Name: Create Application (DESFIRE®)
Description: This function is used to create a new application in the card. The application is identified by its Application Identifier on 24 bits. The Application ID “000000” is reserved as a reference to the PICC itself. Depending on the PICC master key settings a preceding Master Key authentication may be required

CLASS	INS	DATA IN		
$11	**$01**	**AppID**	**Opt**	**KeyNum**

AppID: 3 bytes : ID number of the App in the card (Msb first)
Opt: 1 byte : Or Bit between the following options:
xxxx0001b if Configuration is changeable
xxxx0010b if create/delete operation are Free (without master key)
xxxx0100b if access to list directory is free (without master key)
xxxx1000b if master key setting can be changed
KeyNum: 1 byte : Key number of key usable for that new application

Returned response format

CLASS	INS	DATA OUT	
$11	**$01**	**Length**	**Status**

Length: 1 byte : Response length $02 for successful operation
Status: 2 bytes : See List of response code

$11 $02

Devices: *all printer models with ASK module*

Name: Delete Application (DESFIRE®)
Description: This function is used to permanently deactivate application on the PICC if that operation is allowed. Depending on the PICC master key settings a preceding Master Key authentication may be required

CLASS	INS	DATA IN
$11	**$02**	**AppID**

AppID: 3 bytes : ID number of the App in the card

Returned response format

CLASS	INS	DATA OUT	
$11	**$02**	**Length**	**Status**

Length: 1 byte : Response length $02 for successful operation
Status: 2 bytes : See List of response code

$11 $03

Devices: *all printer models with ASK module*

Name: Select Application (DESFIRE®)
Description: This function is used to select one application for further access in the card.

CLASS	INS	DATA IN
$11	$03	AppID

AppID: 3 bytes : ID number of the App in the card

Returned response format

CLASS	INS	DATA OUT	
$11	$03	Length	Status

Length: 1 byte : Response length $02 for successful operation
Status: 2 bytes : See List of response code

$11 $04

Devices: *all printer models with ASK module*

Name: Format PICC (DESFIRE®)
Description: This function is used to format Picc(card) File system.
N.B.: That command needs to be passed on after an authentication.

CLASS	INS	DATA IN
$11	$04	-

AppID: 3 bytes : ID number of the App in the card

Returned response format

CLASS	INS	DATA OUT	
$11	$04	Length	Status

Length: 1 byte : Response length $02 for successful operation
Status: 2 bytes : See List of response code

$11 $05

Devices: *all printer models with ASK module*

Name: Get Application IDs (DESFIRE®)
Description: This function is used to retrieve the current Application ID.

CLASS	INS	DATA IN
$11	$05	NbID

NbID: 1 byte : Number of ID to return (0 to 32)

Returned response format

CLASS	INS	DATA OUT		repeated n times
$11	$05	Length	Status	IDs (3 bytes)

Length: 1 byte : Response length $02 +nx3 for successful operation
Status: 2 bytes : See List of response code
IDs: nx3 bytes: Id for each application found

$11 $06

Devices: *all printer models with ASK module*

Name: Get version (DESFIRE®)
Description: This function returns the version of the Picc firmware.

CLASS	INS	DATA IN
$11	$06	-

Returned response format

CLASS	INS	DATA OUT							
$11	$06	Length	Status	HardInfo	SoftInfo	UID	Batch	Cw	Year

Length: 1 byte : Response length $1E for successful operation
Status: 2 bytes : See List of response code
HardInfo: 7 bytes:
byte 1: code of the vendor
byte 2; code of the type
byte 3: code of the subtype
byte 4: code of the major version number
byte 5: code of the minor version number
byte 6: code of the storage size
byte 7: code of the communication protocol
SoftInfo: 7 bytes:
byte 1: code of the vendor
byte 2; code of the type
byte 3: code of the subtype
byte 4: code of the major version number
byte 5: code of the minor version number
byte 6: code of the storage size
byte 7: code of the communication protocol
UID: 7bytes: unique serial number
Batch: 5bytes: production batch number
Cw: 1 byte: calendar year of prod.
Year: 1 byte: year of manufacturing

$11 $07

Devices: *all printer models with ASK module*

Name: Get Free Mem (DESFIRE®)
Description: This function is used to retrieve the size available on the PICC.

CLASS	INS	DATA IN
$11	$07	-

Returned response format

CLASS	INS	DATA OUT	
$11	$07	Length	Status

repeated n times

Size (3 bytes)

Length: 1 byte : Response length $02 +3 for successful operation
Status: 2 bytes : See List of response code
Size: 3 bytes: Size of free memory available.

$11 $07

Devices: *all printer models with ASK module*

Name: Authenticate (DESFIRE®)
Description: This function returns the size of the free memory in the Picc firmware.

CLASS	INS	DATA IN
$11	$08	KeyNum

KeyNum: 1 byte : KeyNumber : specify the number of the access key which will be used for the authentication. Depending on this key number and the ID of the currently selected application KeyNum will address either an ordinary application key, the master key or even the PICC master key.

ID OF THE CURRENTLY SELECTED APPLICATION	KEYNUM	KEY USED FOR AUTHENTICATION
not 0	Not 0	Normal application key
not 0	0	Application master key
0	Not 0	Not allowed
0	0	PICC master key

If the PICC level is currently selected (application ID is 0), KeyNum must be 0.
If an application is currently selected (application ID is not 0) the given KeyNum must be in the range from 0 to number of keys created with the application.

Returned response format

CLASS	INS	DATA OUT	
$11	$08	Length	Status

Length: 1 byte : Response length $02 for successful operation
Status: 2 bytes : See List of response code

$11 $09

Devices: *all printer models with ASK module*

Name: Commit Transaction (DESFIRE®)
Description: This function commits the transaction to end a transaction operation with changes.

CLASS	INS	DATA IN
$11	$09	-

Returned response format

CLASS	INS	DATA OUT	
$11	$09	Length	Status

Length: 1 byte : Response length $02 for successful operation
Status: 2 bytes : See List of response code

$11 $0A

Devices: *all printer models with ASK module*

Name: Abort Transaction (DESFIRE®)
Description: This function aborts the current transaction to end a transaction operation with no changes.

CLASS	INS	DATA IN
$11	$0A	-

Returned response format

CLASS	INS	DATA OUT	
$11	$0A	Length	Status

Length: 1 byte : Response length $02 for successful operation
Status: 2 bytes : See List of response code

$11 $0B

Devices: *all printer models with ASK module*

Name: Change Key (DESFIRE®)
Description: This function allow to change any key stored on the PICC. If AID 00 00 00 is selected, the change applies to the Master key and therefore only the key N°00 is valid.

CLASS	INS	DATA IN
$11	$0B	KeyNum

KeyNumber: 1 byte : specify the number of the access key which shall be changed Depending on this key number and the ID of the currently selected application, KeyNum will address either an ordinary application key, the master key or even the PICC master key.

ID OF THE CURRENTLY SELECTED APPLICATION	KEYNUM	KEY USED FOR AUTHENTICATION
not 0	Not 0	Normal application key
not 0	0	Application master key
0	Not 0	Not allowed
0	0	PICC master key

If the PICC level is currently selected (application ID is 0), KeyNum must be 0.
If an application is currently selected (application ID is not 0) the given KeyNum must be in the range from 0 to number of keys created with the application.

Returned response format

CLASS	INS	DATA OUT	
$11	$0B	Length	Status

Length: 1 byte : Response length $02 for successful operation
Status: 2 bytes : See List of response code

$11 $0C

Devices: *all printer models with ASK module*

Name: Change Key Settings (DESFIRE®)
Description: This function changes the key settings information if possible.

CLASS	INS	DATA IN
$11	$0C	NewKeySetting

NewKeySetting: 1 byte : specify the new master key settings either for the currently selected application or for the whole PICC depending whether an application is currently selected (Application ID is non zero) or the PICC level is selected (application ID is 0)

Returned response format

CLASS	INS	DATA OUT	
$11	$0C	Length	Status

Length: 1 byte : Response length $02 for successful operation
Status: 2 bytes : See List of response code

$11 $0D

Devices: *all printer models with ASK module*

Name: Get Key Settings (DESFIRE®)
Description: This function gets the configuration information on the PIDD and the application master key configuration settings.

CLASS	INS	DATA IN
$11	$0D	-

Returned response format

CLASS	INS	DATA OUT			
$11	$0D	Length	Status	KeySetting	Nkeys

Length: 1 byte : Response length $02 for successful operation
Status: 2 bytes : See List of response code
KeySetting: 1 byte : key settings either for the currently selected application or for the whole PICC depending whether an application is currently selected (Application ID is non zero) or the PICC level is selected (application ID is 0)
NKeys: 1 byte : Number of keys defined for the current selected application or for the whole PICC.

$11 $0E

Devices: *all printer models with ASK module*

Name: Get Key Version (DESFIRE®)

CLASS	INS	DATA IN
$11	$0E	KeyNumber

KeyNumber: 1 byte : specify the number of the access key

Returned response format

CLASS	INS	DATA OUT		
$11	$0E	Length	Status	KeyVersion

Length: 1 byte : Response length $02 for successful operation
Status: 2 bytes : See List of response code
KeyVersion: 1 byte : version returned

$11 $10

Devices: *all printer models with ASK module*

Name: Change File Settings (DESFIRE®)
Description: This function changes the file configuration on the PICC.

CLASS	INS	DATA IN			
$11	$10	FileID	CommEncrypted	NewCommMode	NewAccessRights

FileID: 1 byte: This is the ID of the file whose communication mode and access rights settings shall be changed.
CommEncrypted: 1 byte: if non 0, encrypt the communication
NewCommMode: 1 byte: new communication mode
NewAccessRights: 1 byte: specify the access right setting for this file

Returned response format

CLASS	INS	DATA OUT	
$11	$10	Length	Status

Length: 1 byte : Response length $02 for successful operation
Status: 2 bytes : See List of response code

$11 $11

Devices: *all printer models with ASK module*

Name: Clear Record File (DESFIRE®)
Description: This function clears the record files selected by the input param.

CLASS	INS	DATA IN
$11	$11	FileID

FileID: 1 byte: This is the ID of the file which shall be cleared.

Returned response format

CLASS	INS	DATA OUT	
$11	$11	Length	Status

Length: 1 byte : Response length $02 for successful operation
Status: 2 bytes : See List of response code

$11 $10

Devices: *all printer models with ASK module*

Name: Create Backup Data File (DESFIRE®)
Description: Creation of a Backup Data File.

CLASS	INS	DATA IN			
$11	$12	FileID	CommMode	AccessRights	FileSize

FileID: 1 byte: This is the ID of the file for which the new Backup Data File is to be created
CommMode: 1 byte: File communication mode
AccessRights: 2 bytes: New File access rights settings
FileSize: 3 bytes: Size of the new Backup Data File

Returned response format

CLASS	INS	DATA OUT	
$11	$12	Length	Status

Length: 1 byte : Response length $02 for successful operation
Status: 2 bytes : See List of response code

$11 $13

Devices: *all printer models with ASK module*

Name: Create Cyclic Record File (DESFIRE®)

CLASS	INS	DATA IN				
$11	$13	FileID	CommMode	AccessRights	RecordSize	MaxNRecord

FileID: 1 byte: This is the ID of the file for which the new Cyclic record File is to be created
CommMode: 1 byte: File communication mode
AccessRights: 2 bytes: New File access rights settings
RecordSize: 3 bytes: Size of the new Cyclic File in bytes
MaxNRecord: 3 bytes: Number of the records for the new Cyclic File

Returned response format

CLASS	INS	DATA OUT	
$11	$13	Length	Status

Length: 1 byte : Response length $02 for successful operation
Status: 2 bytes : See List of response code

$11 $14

Devices: *all printer models with ASK module*

Name: Create Linear Record File (DESFIRE®)
Description: Creation of a Linear Data File.

CLASS	INS	DATA IN				
$11	$14	FileID	CommMode	AccessRights	RecordSize	MaxNRecord

FileID: 1 byte: This is the ID of the file for which the new Linear record File is to be created
CommMode: 1 byte: File communication mode
AccessRights: 2 bytes: New File access rights settings
RecordSize: 3 bytes: Size of the new linear File in bytes
MaxNRecord: 3 bytes: Number of the records for the new linear File

Returned response format

CLASS	INS	DATA OUT	
$11	$14	Length	Status

Length: 1 byte : Response length $02 for successful operation
Status: 2 bytes : See List of response code

$11 $15

Devices: *all printer models with ASK module*

Name: Create Standard Data File (DESFIRE®)
Description: Creation of a Standard Data File.

CLASS	INS	DATA IN			
$11	$15	FileID	CommMode	AccessRights	FileSize

FileID: 1 byte: This is the ID of the file for which the new File is to be created
CommMode: 1 byte: File communication mode
AccessRights: 2 bytes: New File access rights settings
FileSize: 3 bytes: Size of the new File in bytes

Returned response format

CLASS	INS	DATA OUT	
$11	$15	Length	Status

Length: 1 byte : Response length $02 for successful operation
Status: 2 bytes : See List of response code

$11 $16

Devices: *all printer models with ASK module*

Name: Create Value File (DESFIRE®)
Description: Creation of a Value File.

CLASS	INS	DATA IN						
$11	$16	FileID	CommMode	AccessRights	Lower	Upper	Initial	Limited

FileID: 1 byte: This is the ID of the file for which the new File is to be created
CommMode: 1 byte: File communication mode
AccessRights: 2 bytes: New File access rights settings
Lower: 4 bytes: Min amount for the value file
Upper: 4 bytes: Max amount for the value file
Initial: 4 bytes: amount with which the value file will be created
Limited: 1 bytes: Specifies wheter the limited credit command is enabled (non 0) for the new value file or not.

Returned response format

CLASS	INS	DATA OUT	
$11	$16	Length	Status

Length: 1 byte : Response length $02 for successful operation
Status: 2 bytes : See List of response code

$11 $17

Devices: *all printer models with ASK module*

Name: Credit (DESFIRE®)
Description: Credit a value on a value File.

CLASS	INS	DATA IN		
$11	$17	FileID	CommMode	Amount

FileID: 1 byte: This is the ID of the file for which the new File is to be credited
CommMode: 1 byte: File communication mode
Amount: 4 bytes: amount to be credited in the value file

Returned response format

CLASS	INS	DATA OUT	
$11	$17	Length	Status

Length: 1 byte : Response length $02 for successful operation
Status: 2 bytes : See List of response code

$11 $18

Devices: *all printer models with ASK module*

Name: Debit (DESFIRE®)
Description: Debit a value on a value File.

CLASS	INS	DATA IN		
$11	$18	FileID	CommMode	Amount

FileID: 1 byte: This is the ID of the file for which the new File is to be credited
CommMode: 1 byte: File communication mode
Amount: 4 bytes: amount to be credited in the value file

Returned response format

CLASS	INS	DATA OUT	
$11	$18	Length	Status

Length: 1 byte : Response length $02 for successful operation
Status: 2 bytes : See List of response code

$11 $19

Devices: *all printer models with ASK module*

Name: Delete File (DESFIRE®)
Description: Delete a File .

CLASS	INS	DATA IN
$11	$19	FileID

FileID: 1 byte: This is the ID of the file for which the new File is to be credited

Returned response format

CLASS	INS	DATA OUT	
$11	$19	Length	Status

Length: 1 byte : Response length $02 for successful operation
Status: 2 bytes : See List of response code

$11 $1A

Devices: *all printer models with ASK module*

Name: Get File IDs (DESFIRE®)
Description: Get File ID for the current application.

CLASS	INS	DATA IN
$11	$1A	MaxFileIDs

MaxFileIDs: 1 byte: Max response expected

Returned response format

CLASS	INS	DATA OUT		
$11	$1A	Length	Status	NbFound

Nb times

FileID

Length: 1 byte : Response length
Status: 2 bytes : See List of response code
NbFound: 1 byte : Number of FileId found
FileId: NbFound*1 bytes : FileID array

$11 $1B

Devices: *all printer models with ASK module*

Name: Get File Settings (DESFIRE®)
Description: Get File Settings.

CLASS	INS	DATA IN
$11	$1B	FileID

FileID: 1 byte: This is the ID of the file for which the setting is to be Retrieve

Returned response format

CLASS	INS	DATA OUT				
$11	$1B	Length	Status	FileType	CommMode	AccessRights

Length: 1 byte : Response length $02 for successful operation
Status: 2 bytes : See List of response code
CommMode: 1 byte: File communication mode
AccessRights: 2 bytes: New File access rights settings

$11 $1C

Devices: *all printer models with ASK module*

Name: Get Value (DESFIRE®)
Description: Get Value.

CLASS	INS	DATA IN	
$11	$1C	FileID	CommMode

FileID: 1 byte: This is the ID of the file for which the setting is to be Retrieve
CommMode: 1 byte: File communication mode

Returned response format

CLASS	INS	DATA OUT		
$11	$1C	Length	Status	Amount

Length: 1 byte : Response length $02 for successful operation
Status: 2 bytes : See List of response code
Amount: 4 bytes: Amount of the value returned

$11 $1D

Devices: *all printer models with ASK module*

Name: Limited Credit (DESFIRE®)
Description: Limited Credit.
N.B.: This command needs a commit_transaction to be validated.

CLASS	INS	DATA IN		
$11	$1D	FileID	CommMode	Amount

FileID: 1 byte: This is the ID of the file for which the credit is to increase
CommMode: 1 byte: File communication mode
Amount: 4 bytes: Max Amount that can be added to the Values'File value.

Returned response format

CLASS	INS	DATA OUT	
$11	$1D	Length	Status

Length: 1 byte : Response length $02 for successful operation
Status: 2 bytes : See List of response code

$11 $1E

Devices: *all printer models with ASK module*

Name: Read Data (DESFIRE®)
Description: Read Data.

CLASS	INS	DATA IN			
$11	$1E	FileID	CommMode	FromOffset	NbByteToRead

FileID: 1 byte: This is the ID of the file for which the setting is to be Retrieve
CommMode: 1 byte: File communication mode
FromOffset: 2 bytes: Offset in the File
NbByteToRead 2 bytes: Nb byte to read

Returned response format

CLASS	INS	DATA OUT			
$11	$1E	Length	Status	NbByteToRead	Data

Length: 2 bytes : Response length
Status: 2 bytes : See List of response code
NbByteRead 2 bytes: Nb Bytes read
Data: NbByteRead* bytes: Data read in the File

$11 $1F

Devices: *all printer models with ASK module*

Name: Read Record (DESFIRE®)
Description: Read Record.

CLASS	INS	DATA IN				
$11	$1F	FileID	CommMode	FromRecord	NbRecordToRead	RecordSize

FileID: 1 byte: This is the ID of the file for which the setting is to be Retrieve
CommMode: 1 byte: File communication mode
FromRecord: 2 bytes: Record number from which Data are read
NbRecordToRead 2 bytes: Number of record to read
RecordSize 2 bytes: record size

Returned response format

CLASS	INS	DATA OUT			
$11	$1F	Length	Status	NbRecordRead	Data

Length: 2 bytes : Response length
Status: 2 bytes : See List of response code
NbRecordRead 2 bytes: Nb Bytes read
Data: NbRecordRead*Record Size* bytes: Data read in the File

$11 $20

Devices: *all printer models with ASK module*

Name: Write Data (DESFIRE®)
Description: Write Data.

CLASS	INS	DATA IN				
$11	$20	FileID	CommMode	FromOffset	NbByteToWrite	Data

FileID: 1 byte: This is the ID of the file for which the setting is to be Retrieve
CommMode: 1 byte: File communication mode
FromOffset: 2 bytes: Offset in the File
NbByteToWrite 2 bytes: Nb byte to write
Data NbByteToWrite* bytes: Data to write

Returned response format

CLASS	INS	DATA OUT	
$11	$20	Length	Status

Length: 1 byte : Response length $02 for successful operation
Status: 2 bytes : See List of response code

$11 $21

Devices: *all printer models with ASK module*

Name: Write Record (DESFIRE®)
Description: Write Record.

CLASS	INS	DATA IN				
$11	$21	FileID	CommMode	FromOffset	NbRecordToWrite	Data

FileID: 1 byte: This is the ID of the file for which the setting is to be Retrieve
CommMode: 1 byte: File communication mode
FromOffset: 2 bytes: Record number from which Data are written
NbRecordToWrite 2 bytes: Number of record to write
Data NbRecordToWrite*bytes: Data To Write

Returned response format

CLASS	INS	DATA OUT	
$11	$21	Length	Status

Length: 1 byte : Response length $02 for successful operation
Status: 2 bytes : See List of response code

$11 $22

Devices: *all printer models with ASK module*

Name: SAM Get Version (DESFIRE®)
Description: This function returns the SAM firmware info.

CLASS	INS	DATA IN
$11	$22	-

Returned response format

CLASS	INS	DATA OUT		
$11	$22	Length	Status	SAM VERSION

Length: 1 byte : Response length $20 for successful operation
Status: 2 bytes : See List of response code
SAM VERSION: 30 Bytes: see spec

$11 $23

Devices: *all printer models with ASK module*

Name: SAM Select Application (DESFIRE®)
Description: This function selects an application in the SAM.

CLASS	INS	DATA IN
$11	$23	DF AID

DFAID: 3Bytes: Directory File AID

Returned response format

CLASS	INS	DATA OUT	
$11	$23	Length	Status

Length: 1 byte : Response length 0x02
Status: 2 bytes : See List of response code

$11 $24

Devices: *all printer models with ASK module*

Name: SAM Load Init Vector (DESFIRE®)
Description: This function load an init vector in the SAM for 3DES seeding.

CLASS	INS	DATA IN
$11	**$24**	**Init Vector**

InitVector: 8 bytes : crypto seed

Returned response format

CLASS	INS	DATA OUT	
$11	**$24**	**Length**	**Status**

Length: 1 byte : Response length 0x02
Status: 2 bytes : See List of response code

$11 $25

Devices: *all printer models with ASK module*

Name: SAM Get Key Entry (DESFIRE®)
Description: This function returns the key entry info.

CLASS	INS	DATA IN
$11	**$25**	**Key Num**

Key Num: 1Byte: number of the key entry to be returned

Returned response format

CLASS	INS	DATA OUT		
$11	**$25**	**Length**	**Status**	**Key Entry**

Length: 1 byte : Response length $05 for successful operation
Status: 2 bytes : See List of response code
Key Entry: 3 bytes: 3 key versions

$11 $26

Devices: *all printer models with ASK module*

Name: SAM Get Kuc Entry (DESFIRE®)
Description: This function returns the Kuc info.

CLASS	INS	DATA IN
$11	$26	RefNumKuc

RefNumKuc: 1 byte : key usage counter entry reference Number

Returned response format

CLASS	INS	DATA OUT		
$11	$25	Length	Status	Kuc Entry

Length: 1 byte : Response length
Status: 2 bytes : See List of response code
KucEntry: x bytes: see spec

$11 $26

Devices: *all printer models with ASK module*

Name: SAM Disable crypto(DESFIRE®)
Description: This function disables the crypto function of certain part/function on the SAM/PICC.

CLASS	INS	DATA IN
$11	$26	PROMAS

PROMAS: 2 bytes : See explication below

The NV. Programming bit mask (PROMAS) 2 bytes parameters is defined as follow

bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Disable Generate MAC	Disable Verify MAC	Disable Encryption	Disable Decryption	Disable SAM changeKey_PICC	RFU=0										

Returned response format

CLASS	INS	DATA OUT	
$11	$26	Length	Status

Length: 1 byte : Response length 0x02
Status: 2 bytes : See List of response code

3.7.12 FELICA® Class (N°= $12) (NOT IMPLEMENTED YET)

N.B. this class is not implemented yet.
The FELICA® Class is based on the Sony Felica® contactless protocol, it allows communicating and managing the FELICA® card or a NFC Device embedding the Felica® protocol.

3.7.12.1 Felica® Cards Memory organization

The unit of data access is one block (16 bytes).

The maximum number of simultaneously accessible blocks is 12 blocks (176 Bytes) in read mode.
The maximum number of simultaneously accessible blocks is 8 blocks (128 Bytes) in write mode.

Due to the multi-application usage concept the Felica card employs two concepts called "area" and "service"
The area serves as a security firewall between different providers sharing the same card.
The service is used to define the method of accessing the memory.

User access to the memory is performed via a 2-byte code called the area code or service code.
The memory space is divided into user block and system block areas.
The user block area is used to write user data. It can be allocated by defining the service.
In addition to card-unique information, the system block area contains information about the card structure such as Area
Definition Block(s) and Service Definition Block(s).

Category	Block Type Block Attribute		Description
System Block	Manufacture ID Block		Provide manufacture information
	Issue ID Block		Provide issuance information
	System Definition Block		Provide definition of the entire system
	Area 000 Definition Block		Definition of the highest hierarchical level
	Area Definition Block		Definition of hierarchy area used to control services
	Service Definition Block		Provide definition of the services
User Block	Random Access Block	Read/Write Access Read/Write Access (w/o security) Read Only Acces Read Only Access (w/o security)	Enable Access using block numbers
	Cyclic Access Block	Read/Write Access Read/Write Access (w/o security) Read Only Access Read Only Access (w/o security)	Write: Every time data is written, renumbers data blocks from the newest one. (Prevents same data from being written twice.) Read: Allows block selection by its number.
	Purse Block	Direct Access Direct Access (w/o security) Cash Back/Decrement Access Cash Back/Decrement Access (w/o security) Decrement Access Decrement Access (w/o security) Read Only Access Read Only Access (w/o security)	Allows access using the block number. Using the Execute ID prevents the same processing from being repeated in the retry process.

3.7.12.2 Felica® Cards Free Access Commands

Polling Command

Polling is embedded in the ENTER Hunt Phase system command.
NB.: The Sytem code parameter (2 bytes) and the TimeSlot parameter (1 Byte) are part of the 01_17_Enter_Hunt_Phase_Parameter command with default values of ffffh and 03h respectively. Used to identify the card from the Reader/Writer.

Request Service Command

Specifies the area/service and obtains the area/service key version data when the specified area/service resides in the card.

Request Response Command

Used to check the existence and status of the card.

Read Without Encryption Command

In Mode 0, used to read card data in block units.

Write Without Encryption Command

In Mode 0, used to write data into the card memory in block units.

Search Service Code Command

Used to obtain the service code (or the area code and end service code) of the area/service registered in the card.

Request System Code Command

Allows searching for the card's registered system code to acquire the system code list.

3.7.12.3 Felica® Cards Secure Access Commands

Authentication1

Enables the Reader/Writer to authenticate the card.

Authentication2

Enables the card to authenticate the Reader/Writer.

Read

Used to read card data in block units, using the service specified in the authentication process.

Write

Used to write data to the card in block units, using the service specified in the authentication process.

Other command exists that require authentication for personalization but are not described in this document.

3.7.12.4 Error Code List

Cod	Status	Detail
$00	No error	Normal Termination
$01	Purse Error	Lack of balance (in the decrement access mode) or data overflow (in the cash back access mode) detected during the use of the purse service.
$02	Cash Back Error	In the purse cash back mode, the immediately preceding decrement value is smaller than the cash back data.
$70	Memory Error	Cannot write in the memory. CRC error or garbled bits detected in the memory.
$71	Excessive Writes	The memory has been written more than 0x1FFFF (131071) times. (Processing continues, as this is not considered an error.)
$A1	Incorrect Service Number	The service number is not correctly specified on the service code list when using the Read Without Encryption or Write Without Encryption command.
$A2	Incorrect Block Number	The block number is not correctly specified in the command used for reading, writing or issuing a card.
$A3	Incorrect Service Assignment	The order of services specified on the block list is incorrect.
$A4	Incorrect Service Type	The service type is incorrect, making it impossible to register an area or service.
$A5	Incorrect Access	Implies one of the following error messages.
		- Do not attempt to create a child area below the area that does not allow creation of such areas.
		- Do not try to write using the read only service.
		- Do not write IDm twice in the same card.
		- Do not use Read or write Without Encryption commands for service requiring encryption.
		- Do not attempt to divide the card that has already been logically divided.
$A6	Incorrect Service Code	The service being accessed is non-existent. The service cannot be registered because the service code differs inside and outside the package.
$A7	Incorrect Access Mode	The access mode setting in the block list is incorrect. The key information cannot be changed by using the Write Without Encryption command.
$A8		Incorrect Record Number A record with a number larger than the service size is being accessed.
$A9		Change System Failure The Change System Block command cannot be executed because the previous operation by a registration-related command or Separate command is not completed successfully.
$AA	Key Change Failure	The key cannot be changed because of incorrect key change information.
$AB	Incorrect Package Parity	The package parity of the command used for issuing the card is incorrect.
$AC	Incorrect Parameter	Registration is impossible because the scope of service codes is inconsistent between the parent area and the area or service to be newly registered.
$AD	Double Registration of Same Service	Cannot register the same area or service twice.
$AE	Incorrect System Code	The system code of the Register Issue ID command is neither FFFF nor the same as the system code to be registered.
$AF	Too Many Simultaneous Cyclic Writes	The number of blocks being written simultaneously is larger than the number of records available for the cyclic service. (This error message is not displayed when the blocks to be written do not appear consecutively on the block list.)
$FD	Bad IDm	The IDm replyied does not match with the IDm required in the command
$FE	Card Timeout	No card response
$FF	SAM Timeout	No SAM response

3.7.12.5 Reader functions

$12 $01

Devices: *all printer models with ASK module*

Name: REQ SERVICE (FREE FELICA® ACCESS)
Description: Specify Area/service and obtain the Area/Service Key version data when it exists on card.

CLASS	INS	DATA IN	
$12	$01	Length	ASKList

Length: 1 byte: n Number of Area/Services
ASKList: 2*n bytes: Area/Service Key version list

Returned response

CLASS	INS	DATA OUT		
$12	$01	Error	Length	ASKList

Error: 1 byte: See Error code List
Length: 1 byte: n Number of existing Area/Service return by the card among the ones requested
ASKList: 2*n bytes: Area/Service Key version list

$12 $02

Devices: *all printer models with ASK module*

Name: REQ RESPONSE (FREE FELICA® ACCESS)
Description: Control that the card still exists and gives its Mode parameter.

CLASS	INS	DATA IN
$12	$02	-

Returned response

CLASS	INS	DATA OUT	
$12	$02	Error	Mode

Error: 1 byte: See Error code List
Mode: 1 byte: internal Mode (From $0 to $3)

$12 $03

Devices: *all printer models with ASK module*

Name: READ WITHOUT ENCRYPTION (FREE FELICA® ACCESS)
Description: Used to read card data in block units (16 bytes) without encryption.

CLASS	INS	DATA IN			
$12	**$03**	**Service Length**	**Service List**	**Block Length**	**Block list**

Service Length: 1 byte: m Number of Service
Service List: 2m to 3m bytes: Service list
Block Length: 1 byte: n Number of blocks
Block List: 2n to 3n bytes: Block list

Returned response

CLASS	INS	DATA OUT		
$12	**$03**	**Error**	**Length**	**Data**

Error: 1 byte: See Error code List
Length: 1 byte: n Number block Data read
Data: 16*n bytes: Data read

$12 $04

Devices: *all printer models with ASK module*

Name: WRITE WITHOUT ENCRYPTION (FREE FELICA® ACCESS)
Description: Used to read card data in block units (16 bytes) without encryption.

CLASS	INS	DATA IN				
$12	**$04**	**Service Length**	**Service List**	**Block Length**	**Block list**	**Block Data**

Service Length: 1 byte: m Number of Services
Service List: 2m to 3m bytes: Service list
Block Length: 1 byte: n Number of blocks
Block List: 2n to 3n bytes: Block list
Block Data: 16m bytes Block data to write

Returned response

CLASS	INS	DATA OUT
$12	**$04**	**Error**

Error: 1 byte: See Error code List

$12 $05

Devices: *all printer models with ASK module*

Name: SEARCH SERVICE CODE (FREE FELICA® ACCESS)
Description: Obtain the code of area and service from the card.

CLASS	INS	DATA IN
$12	$05	ServiceIndex

ServiceIndex 2 bytes: Service index of search

Returned response

CLASS	INS	DATA OUT		
$12	$05	Error	Area/Service Code	End Service Code

Error: 1 byte: See Error code List
Area/Service code: 2 byte: Area replied
End Service code: 2 byte: in case of area

$12 $06

Devices: *all printer models with ASK module*

Name: MUTUAL AUTHENTICATION (Secure FELICA® ACCESS)
Description: Enables the Reader/Writer to authenticate the card and card to authenticate the Reader/Writer

CLASS	INS	DATA IN
$12	$06	-

Returned response

CLASS	INS	DATA OUT
$12	$06	Error

Error: 1 byte: See Error code List

$12 $07

Devices: *all printer models with ASK module*

Name: READ (Secure FELICA® ACCESS)
Description: Used to read card data in block units, using the service specified in the authentication process.

CLASS	INS	DATA IN	
$12	**$07**	**Number of block**	**Block List**

Number of Block 1 byte n
Block List 2n bytes List of block to read

Returned response

CLASS	INS	DATA OUT	
$12	**$07**	**Error**	**Block data**

Error: 1 byte: See Error code List
Block Data: 16n bytes Data read

$12 $08

Devices: *all printer models with ASK module*

Name: WRITE (Secure FELICA@ ACCESS)
Description: Used to write data to the card in block units, using the service specified in the authentication process.

CLASS	INS	DATA IN		
$12	**$08**	**Number of block**	**Block List**	**Block Data**

Number of Block 1 byte n
Block List 2n bytes List of block to Write
Block Data 16n bytes block Data to write

Returned response

CLASS	INS	DATA OUT
$12	**$08**	**Error**

Error: 1 byte: See Error code List

3.8 USE EXAMPLES

3.8.1 Class CTS

class 6 functions in order to do an UPDATE_CTS command

..... CSC found : reset ASK CSC

Message : 80 02 06 01 00 55 b3

Description: Type command Sens Host -> CSC
..... Command: Active CTS
..... Timing= 0 s 20 ms

Response : 01 0c 06 01 0a 0f 50 02 00 93 96 67 00 00 00 2d

Description: Type command Sens CSC -> Host
..... Function class = class CTS
..... Response : Active CTS
..... Ticket CTS active

<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<

Message : 80 04 06 02 02 02 00 27 9e

Description: Type command Sens Host -> CSC
..... Command : Read CTS
..... ADD: 02
..... NB : 02
..... Timing= 0 s 15 ms

Response : 01 06 06 02 03 02 00 10 00 c7 30

Description: Type command Sens CSC -> Host
..... Function class = class CTS
..... Response : Read CTS
..... Read command success
..... Data read : 00 10

<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<

Message : 80 04 06 02 0a 04 00 35 0c

Description: Type command Sens Host -> CSC
..... Command : Read CTS
..... ADD: 0a
..... NB : 04
..... Timing= 0 s 20 ms

Response : 01 08 06 02 05 02 ff 77 77 77 00 a8 85

Description: Type command Sens CSC -> Host
..... Function class = class CTS
..... Response : Read CTS
..... Read command success
..... Data read : ff 77 77 77

<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<

Message : 80 08 06 03 0a 02 55 44 ee ee 00 76 66

Description: Type command Sens Host -> CSC
..... Command : Update CTS
..... NB : 0a
..... ADD: 02
..... Data to write : 55 44 ee ee 00 76 66 cc cc cc
..... Data read : cc cc cc cc cc cc cc cc cc cc ..... Timing= 0 s 26 ms

Response : 01 06 06 03 03 02 55 44 00 25 3a

Description: Type command Sens CSC -> Host
..... Function class = class CTS
..... Response : Update CTS
..... Update command success
..... Data read : 55 44

<<

Message : 80 04 06 02 0a 04 00 35 0c

Description: Type command Sens Host -> CSC
..... Command : Read CTS
..... ADD: 0a
..... NB : 04
..... Timing= 0 s 20 ms

Response : 01 08 06 02 05 02 aa bb 77 77 00 73 7a

Description: Type command Sens CSC -> Host
..... Function class = class CTS
..... Response : Read CTS
..... Read command success
..... Data read : aa bb 77 77

<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<

3.8.2 Variable class mapping for GTML card

class 5 function : APPEND_RECORD command (in session mode)

..... CSC found : reset ASK CSC

Message : 80 02 01 01 00 50 3f

Description: Type command Sens Host -> CSC
..... Function class = system class
..... Command : CSC version
..... Timing= 0 s 15 ms

Response : 01 41 01 01 43 53 43 20 30 32 2e 30 38 20 3c 70
68 2d 39 36 30 30 2d 31 31 35 32 3e 20 4d 61 72
20 31 33 20 32 30 30 31 20 31 38 3a 30 34 3a 31
34 20 28 43 29 20 41 53 4b 20 47 45 4e 5f 33 58
30 20 00 00 91 cd

Description: Type command Sens CSC -> Host
..... Function class = system class
..... Response : CSC version
..... In Ascii :
..... CSC 02.08 <ph-9600-1152> Mar 13 2001 18:04:14 (C) ASK GEN_3X0

<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<

Message : 80 08 01 03 00 00 00 00 01 00 00 17 69

Description: Type command Sens Host -> CSC
..... Function class = system class
..... Command : Short Enter Hunt Phase
..... Antenna number parameter = 00
..... Contact parameter =00 ISOB=00 ISOA=00
..... TICKET parameter =00 RMT=01
..... Timing= 0 s 25 ms

Response : 01 1e 01 03 00 03 19 00 22 17 6c ff 40 3b 6f 00
00 80 5a 08 03 03 00 00 00 00 22 17 6c 82 90 00
00 39 4f

Description: Type command Direction CSC -> Host
..... Function class = system class
..... Response : TAG search
..... Antenna number parameter = 00 (= 0x80 if failed)
..... Mode RMT Choix Ok
..... ATR length = 19
..... Convention type 3b must be to 3B
..... T0 parameter 6f must be to 6F (if there are 2 parameters after)
..... TB1 parameter 00 must be to 00
..... TC1 parameter 00 must be to 00
..... Indicator category 80 must be to 80
..... Parameter nbconstr 5a must be to 5A
..... Type de composant 08
..... Application type 03
..... ROM version 03 00
..... EEPROM Version 00 00
..... Serial number 00 22 17 6c
..... nbstatus parameter 82 must be to 82
..... Status ATR 90 00 must be to90 00
..... ATR parameter OK

<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<

Message : 80 08 05 10 02 08 01 0d 00 00 00 ae 4b

Description: Type command Sens Host -> CSC
..... Function class = class 05
..... Command : Open Secured Session
..... 02 : validation session
..... 08: Fichier Journal de transport sélectionné
..... 01 : Record number
..... Timing= 0 s 55 ms

Response : 01 23 05 10 00 90 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 36 58

Description: Type command Sens CSC -> Host
..... Function class = class 5
..... Correct return
..... Response: Open Secured Session
..... Number of non-ratified applications 00

<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<

Message : 80 0d 05 01 00 08 05 00 00 00 00 00 20 10 0d 00
0c a8

Description: Type command Direction Host -> CSC
..... Function class = class 05
..... Command: Append Record
..... 00: Access in Default mode
..... 08: Fichier Journal de transport sélectionné
..... Timing= 0 s 41 ms

Response: 01 05 05 01 00 90 00 00 d5 64

Description: Type command Sens CSC -> Host
..... Function class = class 5
..... Correct return
..... Response: Append Record

<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<

Message : 80 08 05 08 08 04 20 00 20 10 00 39 93

Description: Type command Sens Host -> CSC
..... Function class = class 05
..... Command: Select File
..... Timing= 0 s 35 ms

Response: 01 1e 05 08 00 90 00 85 17 08 04 04 1d 03 1f 10
10 10 00 03 03 03 00 00 00 00 00 00 00 00 00 00
00 06 22

Description: Type command Sens CSC -> Host
..... Function class = class 5
..... Correct return
..... Response: Select File
..... Détail du FCI :
..... 85 indique un Tag :
..... SID = 08
..... File type 04 =EF
..... Structure type 04 =Circular
..... Byte number for each record = 1d
..... Record number = 03
..... AC : Access condition for command index 0 1f =ALWAYS Command READ RECORD
..... AC : Access condition for command index 1 10 =INCONNU Command UPDATE RECORD
..... AC : Access condition for command index 2 10 =INCONNU Pas de Commande
..... AC : Access condition for command index 3 10 =INCONNU Command APPEND RECORD
..... Key index to use 00 03 03 03
..... File status 00 Fichier Valide Fichier lisible si Invalide
..... Key index KVC123 pour les DF 00 00 00
..... Plancher des compteurs 00 00 00 Plafond 00 00 00

<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<

Message : 80 0d 05 01 00 00 05 00 00 00 00 00 20 10 0d 00
a6 14

Description: Type command Direction Host -> CSC
..... Function class = class 05
..... Command : Append Record
..... 00: Access in Default mode
..... 00: Selected file
..... Timing= 0 s 41 ms

Response : 01 05 05 01 00 90 00 00 d5 64

Description: Type command Sens CSC -> Host
..... Function class = class 5
..... Correct return
..... Response : Append Record

<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<

Message : 80 02 05 11 00 a0 c9

Description: Type command Sens Host -> CSC
..... Function class = class 05
..... Command : Close Secured Session
..... Timing= 0 s 50 ms

Response : 01 05 05 11 00 90 00 00 95 d0

Description: Type command Sens CSC -> Host
..... Function class = class 5
..... Correct return
..... Response : Close Secured Session

3.8.3 Class CD97

class 3 function : APPEND_RECORD command

..... CSC found : reset ASK CSC

Message : 80 02 01 01 00 50 3f

Description: Type command Sens Host -> CSC
..... Function class = system class
..... Command : CSC version
..... Timing= 0 s 15 ms

Response : 01 41 01 01 43 53 43 20 30 32 2e 30 38 20 3c 70
68 2d 39 36 30 30 2d 31 31 35 32 3e 20 4d 61 72
20 31 33 20 32 30 30 31 20 31 38 3a 30 34 3a 31
34 20 28 43 29 20 41 53 4b 20 47 45 4e 5f 33 58
30 20 00 00 91 cd

Description: Type command Sens CSC -> Host
..... Function class = system class
..... Response : CSC version
..... In Ascii :
..... CSC 02.08 <ph-9600-1152> Mar 13 2001 18:04:14 (C) ASK GEN_3X0

<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<

Message : 80 08 01 03 00 00 00 00 01 00 00 17 69

Description: Type command Sens Host -> CSC
..... Function class = system class
..... Command : Short Enter Hunt Phase
..... Antenna number parameter = 00
..... Contact parameter=00 ISOB=00 ISOA=00
..... TICKET parameter=00 RMT=01
..... Timing= 0 s 24 ms

Response : 01 1e 01 03 00 03 19 00 10 5e f3 ff c0 3b 6f 00
00 80 5a 03 01 01 11 02 04 00 10 5e f3 82 90 00
00 03 62

Description: Type command Direction CSC -> Host
..... Function class = system class
..... Response : TAG search
..... Antenna number parameter = 00 (a 80 si en panne)
..... Mode RMT Choix Ok
..... ATR length = 19
..... Convention type 3b must be to 3B
..... T0 parameter 6f must be to 6F indique si 2 param suivants
..... TB1 parameter 00 must be to 00
..... TC1 parameter 00 must be to 00
..... Indicator category 80 must be to 80
..... Parameter nbconstr 5a must be to 5A
..... Type de composant 03
..... Application type 01
..... ROM version 01 11
..... EEPROM Version 02 04
..... Serial number 00 10 5e f3
..... Nbstatus parameter 82 must be to 82

..... Status ATR 90 00 must be to 90 00
..... ATR parameter OK

<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<

Message : 80 0a 03 01 01 08 05 00 00 00 00 00 00 68 43

Description: Type command Sens Host -> CSC
..... Function class = CD97 class
..... Command : Append Record
..... 01: Access in Protected mode
..... 08: Fichier Journal de transport sélectionné
..... Timing= 0 s 100 ms

Response : 01 05 03 01 00 90 00 00 2f 7c

Description: Type command Sens CSC -> Host
..... Function class = CD97 class
..... Correct return
..... Response : Append Record

<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<

Message : 80 08 03 08 08 04 31 00 31 15 00 d3 29

Description: Type command Sens Host -> CSC
..... Function class = CD97 class
..... Command : Select File
..... Timing= 0 s 31 ms

Response : 01 1e 03 08 00 90 00 85 17 00 04 04 1d 01 1f 12
00 12 01 03 01 03 00 00 00 00 00 00 00 00 00 00
00 5c 76

Description: Type command Sens CSC -> Host
..... Function class = CD97 class
..... Correct return
..... Response : Select File
..... Détail du FCI :
..... 85 indique un Tag :
..... SID = 00
..... File type 04 =EF
..... Structure type 04 =Circular
..... Byte number for each record = 1d
..... Record number = 01
..... AC : Access condition for command index 0 1f =ALWAYS (READ RECORD Command)
..... AC : Access condition for command index 1 12 =PROTECTED^SESSION (UPDATE RECORD Command)
..... AC : Access condition for command index 2 00 =NEVER (No Command)
..... AC : Access condition for command index 3 12 =PROTECTED^SESSION (APPEND RECORD Command)
..... Key index to use 01 03 01 03
..... File status 00 Fichier Valide Fichier lisible si Invalide
..... Key index KVC123 for DF 00 00 00
..... Plancher des compteurs 00 00 00 Plafond 00 00 00

<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<

Message : 80 0a 03 01 01 00 05 00 00 00 00 00 00 d4 6e

Description: Type command Sens Host -> CSC
..... Function class = CD97 class
..... Command : Append Record
..... 01: Access in Protected mode
..... 00: Selected file
..... Timing= 0 s 95 ms

Response : 01 05 03 01 00 90 00 00 2f 7c

Description: Type command Sens CSC -> Host
..... Function class = CD97 class
..... Correct return
..... Response : Append Record

<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<

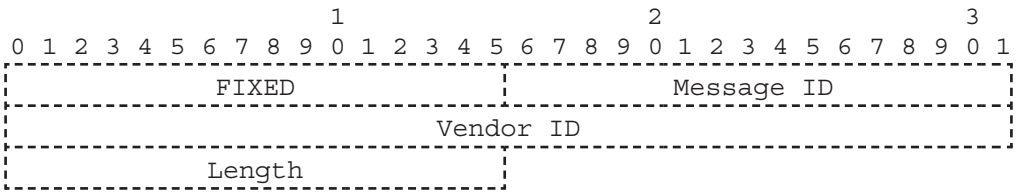
4 UHF RFID COMMANDS

4.1 INTRODUCTION

This chapter describes the message format of the communication protocol used by the host and the reader in order to issuing commands and reply with responses.
The protocol is based on the Attribute Value Pair (AVP) schema and foresees a message header in order to identify the message scope.
The command set and the firmware architecture draw inspiration from the Reader Protocol 1.0 specification draft from EPCGlobal but, at now, this protocol is not fully compatible with the same last specifications.
Message fields are described left to right, with the most significant byte on the left and the least on the right.

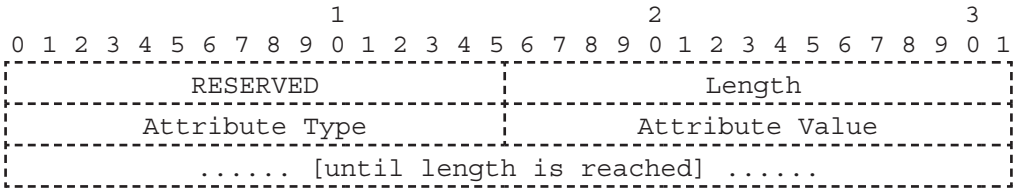
4.2 PROTOCOL SPECIFICATION

CAEN UHF RFID Reader protocol uses two logical communication channels: one for synchronous commands and one for asynchronous notifications. Command channel is mandatory and, at now, it is implemented on top of a TCP/IP socket (port 1000) and on RS232 while notification channels are implemented only with sockets.
All the messages (commands, responses and notifications) are composed by a header and a body. In all cases the body of the message is a list of attribute-value pairs.
Responses always echo the Command AVP sent by the host.
All the packets for the control and notification channel share a common header format:



FIXED: Must be 0x8001 for commands and 0x0001 for responses.
Message ID: Id of the message. It is a sequence number used to map requests to its responses: a request and its corresponding response have the same message ID (the id is local to the channel).
Vendor ID: Must be 21336: the IANA “SMI Network Management Private Enterprise Code” assigned to CAEN SpA.
Length: Encodes the length of the message (in bytes) including the header.

The header is followed by a list of AVPs the number of which depends on the command.
Each AVP have the following format:



RESERVED: The first 16 bits are reserved for future extensions. All reserved bits must be set to 0 on outgoing messages and ignored on incoming messages.
Length: Encodes the length of the AVP packet including the length and the reserved fields.
Attribute type: A 2 byte code identifying the attribute type.
Attribute value: The actual attribute value according to the type. It follows immediately after the Attribute Type field and runs for the remaining bytes indicated in the Length (i.e. Length minus 6 bytes of header).

4.2.1 Attribute types

$01	
Devices:	*all printer models with UHF module*
CommandName:	the command to be executed. All the commands are specified in the relevant table. Attribute value is 2 bytes long.

$02	
Devices:	*all printer models with UHF module*
ResultCode:	a code representing an indication on the result of the command. All the commands are specified in the relevant table. Attribute value is 2 bytes long.

$0E	
Devices:	*all printer models with UHF module*
EventType:	the type of the notified event. Attribute value is 4 bytes long and can assume the following values: 0x00 = Unknown Event 0x01 = Tag glimpsed 0x02 = Tag New 0x03 = Tag Observed 0x04 = Tag Lost 0x05 = Tag Purged

$0F	
Devices:	*all printer models with UHF module*
TagIDLen:	the length of the tag ID. Attribute value is 2 bytes long.

$10	
Devices:	*all printer models with UHF module*
TimeStamp:	an indication of the time. Attribute is 8 bytes long and must be interpreted as follow: - the 4 least significant bytes are the seconds elapsed from the 1 January 1970. - the 4 most significant bytes are the micro-seconds.

$11

Devices: *all printer models with UHF module*

TagID: the ID read from the tag. Attribute value has a maximum length of 12 bytes. For ISO18000 tags only the first 8 bytes are significant while for EPC tags all the 12 bytes are significant.

$12

Devices: *all printer models with UHF module*

TagType: the tag's type. Attribute value is 2 bytes long and can assume the following values:

0x00 = ISO18KB
0x01 = EPCC1G1
0x02 = ISO18KA
0x03 = EPCC1G2
0x05 = EPC119

$1E

Devices: *all printer models with UHF module*

ChannelName: the name of the notification channel. Attribute value has a maximum length of 30 bytes.

$1F

Devices: *all printer models with UHF module*

ChannelAddress: the address of the notification channel. Attribute value has a maximum length of 30 bytes.

$20

Devices: *all printer models with UHF module*

TriggerName: the name of the trigger. Attribute value has a maximum length of 30 bytes.

$21

Devices: *all printer models with UHF module*

TriggerType: the type of the trigger. Attribute value has a maximum length of 30 bytes.

$22

Devices: *all printer models with UHF module*

ReadPointName: a string representing the name of the read point. Attribute value has a maximum length of 5 bytes and can assume the following values:

“Ant0”, “Ant1”, “Ant2”, “Ant3”

$4D

Devices: *all printer models with UHF module*

TagValue: data read from the tag memory (when applicable). Attribute value has a maximum length of 128 bytes.

$4E

Devices: *all printer models with UHF module*

TagAddress: the memory location address of the tag where read or write data (when applicable). Attribute value is 2 bytes long.

$4F

Devices: *all printer models with UHF module*

RESERVED

$50

Devices: *all printer models with UHF module*

Length: a value representing the length of a parameter. Attribute value is 2 bytes long.

$51

Devices: *all printer models with UHF module*

BitRate: a value representing the RF BitRate. Attribute value is 2 bytes long and can assume the following values:

0x0 – Transmit : DSB ASK 10kbit, Receive : FM0 10kbit
0x1 – Transmit : DSB ASK 10kbit, Receive : FM0 40kbit
0x2 – Transmit : DSB ASK 40kbit, Receive : FM0 40kbit
0x3 – Transmit : DSB ASK 40kbit, Receive : FM0 160kbit
0x4 – Transmit : DSB ASK 160kbit, Receive : FM0 400kbit
0x5 – Transmit : DSB ASK 40kbit, Receive : Miller M=2 160kbit
0x6 – Transmit : PR ASK 40kbit, Receive : Miller M=4 250kbit
0x7 – Transmit : PR ASK 40kbit, Receive : Miller M=4 300kbit
0x8 – Transmit : PR ASK 40kbit, Receive : Miller M=2 250kbit

Note: not all the value are supported by all the readers. For the list of mode supported by each reader please refer to the reader's user manual.

$52

Devices: *all printer models with UHF module*

PowerGet: a value representing the RF power. Attribute value is 4 bytes long. (used for read the current setting).

$53

Devices: *all printer models with UHF module*

RESERVED.

$54

Devices: *all printer models with UHF module*

Protocol: a value representing the air protocol. Attribute value is 4 bytes long and can assume the following values:

0x00 = ISO18000-6B
0x01 = EPCC1G1
0x02 = ISO18000-6A
0x03 = EPCC1G2

$56

Devices: *all printer models with UHF module*

ReadPointStatus: a value representing the antenna's status. Attribute value is 4 bytes long and can assume the following values:

0x00 = Bad: antenna is not connected or broken.
0x01 = Poor: antenna has a low quality connection.
0x02 = Good: antenna is well connected.

$57

Devices: *all printer models with UHF module*

Boolean: a value representing a boolean data. Attribute value is 2 bytes long and can assume the following values:
0x00 = FALSE.
Not 0x00 = TRUE.

$58

Devices: *all printer models with UHF module*

IPAddress: a string representing an IP address formatted with the standard IP dotted decimal format. Attribute value has a maximum length of 30 bytes.

$59

Devices: *all printer models with UHF module*

IPNetMask: a string representing an IP netmask formatted with the standard IP dotted decimal format. Attribute value has a maximum length of 30 bytes.

$5A

Devices: *all printer models with UHF module*

IPGateway: a string representing an IP address formatted with the standard IP dotted decimal format. Attribute value has a maximum length of 30 bytes.

$5B

Devices: *all printer models with UHF module*

DESBEnable: used to enable/disable the Data Exchange Status Bit handling for ISO18000-6b and EPC 1.19 anti-collision algorithm. Attribute value is 2 bytes long and can assume the following values:

0x00 = Disable the DESB handling.
Not 0x00 = Enable the DESB handling.

$5C

Devices: *all printer models with UHF module*

FWRelease: a string representing the device's firmware revision. Attribute value has a maximum length of 200 bytes.

$5D

Devices: *all printer models with UHF module*

DESBStatus: used to check the Data Exchange Status Bit handling for ISO18000-6b and EPC 1.19 anti-collision algorithm. Attribute value is 2 bytes long and can assume the following values:

0x00 = DESB handling is not enabled.
Not 0x00 = DESB handling is enabled.

$5E

Devices: *all printer models with UHF module*

EPCPWD: a value representing an EPC tag password. Attribute value is 2 bytes long.

$5F

Devices: *all printer models with UHF module*

RFOnOff: used to start the generation of a continuous wave for test purposes. Attribute value is 2 bytes long and can assume the following vaules:

0x00 = Stop the wave generation.
Not 0x00 = Start the wave generation.

$60

Devices: *all printer models with UHF module*

BaudRate: a value representing the baudrate setting of serial port. Attribute value is 4 bytes long.

$61

Devices: *all printer models with UHF module*

DataBits: a value representing the databits setting of serial port. Attribute value is 4 bytes long.

$62

Devices: *all printer models with UHF module*

StopBits: a value representing the stopbits setting of serial port. Attribute value is 4 bytes long.

$63

Devices: *all printer models with UHF module*

Parity: a value representing the parity setting of serial port. Attribute value is 4 bytes long and can assume the following values:

0x00 = No parity
0x01 = Odd parity
0x02 = Even parity

$64

Devices: *all printer models with UHF module*

FlowCtrl: a value representing the flow control setting of serial port. Attribute value is 4 bytes long and can assume the following values:

0x00 = No flow control
0x01 = Hardware flow control
0x02 = Software flow control (not yet implemented)

$65

Devices: *all printer models with UHF module*

DateTime: a value representing a date and time. Attribute value has a maximum length of 30 bytes. The data format is:

YYYY–MM–DD HH:MM:SS

$66

Devices: *all printer models with UHF module*

SelUnselOp: a value representing the tag selection operation defined by the ISO18000-6B protocol. Attribute value is 2 bytes long and can assume the following values:

0x00 = select equal
0x01 = select not equal
0x02 = select greater than
0x03 = select lower than
0x04 = unselect equal
0x05 = unselect not equal
0x06 = unselect greater than
0x07 = unselect lower than

$67

Devices: *all printer models with UHF module*

Bitmask: a value representing the bitmask parameter of tag selection operations defined by the ISO18000-6B protocol. Attribute value is 2 bytes long (only 8 least significant bits are used).

$68

Devices: *all printer models with UHF module*

REESERVED.

$69

Devices: *all printer models with UHF module*

IORegister: a value representing the status of the I/O lines of the reader. Where input lines are separated from output ones, input lines are mapped on the less significant bits while outputs are mapped on the most significant. Attribute value is 4 bytes long (effective used bits depend on the reader model).

$6A

Devices: *all printer models with UHF module*

ConfigParameter: a value representing a configuration parameter. Attribute value is 4 bytes long and can assume the following values:

0x00 = ReadCycle configuration
0x01 = Observed Threshold configuation
0x02 = Lost Threshold configuration
0x03 = Starting Q value (Valid values : 0 ÷ 15). EPC C1GEN2 Protocol only.
0x04 = Session (Valid values : 0 ÷ 3). EPC C1GEN2 protocol only.
0x05 = Target (Valid values : 0 ÷ 1). EPC C1GEN2 protocol only.
0x06 = Selected (Valid values : 0 ÷ 1). EPC C1GEN2 protocol only.
0x07 = Data Exchange Status B (Valid values : 0 ÷ 1). ISO 18000-6B protocol only.

$6B

Devices: *all printer models with UHF module*

ConfigValue: a value for the configuration parameter. Attribute value is 4 bytes long.

$6C

Devices: *all printer models with UHF module*

NoOfTriggers: a value representing the number of triggers. Attribute value is 2 bytes long.

$6D

Devices: *all printer models with UHF module*

NoOfChannels: a value representing the number of channels. Attribute value is 2 bytes long.

$6E

Devices: *all printer models with UHF module*

EventMode: a value representing the event handling mode. Attribute value is 2 bytes long and can assume the following values:

0x00 = ReadCycle mode
0x01 = Time Mode
0x02 = No Event Mode

$6F

Devices: *all printer models with UHF module*

UpgradeType: a value representing the type of upgrade to perform. Attribute value is 2 bytes long and can assume the following values:

0x01 = TFTP firmware upgrade.

$70

Devices: *all printer models with UHF module*

UpgradeArgument: a value representing the argument for the requested upgrade. Attribute value has a maximum length of 255 bytes.
For TFTP upgrade (code 0x01) the string has the form:
'<tftpserverip>:<filename>'.

$71

Devices: *all printer models with UHF module*

MemoryBank: a value representing the memory bank of a EPC Class 1 Generation 2 tag. Attribute value is 2 bytes long and can assume the following values:

0x00 = Reserved Memory Bank
0x01 = EPC Memory Bank
0x02 = TID Memory Bank
0x03 = User Memory Bank

$72

Devices: *all printer models with UHF module*

Payload: a value representing the payload parameter for the EPC Class 1 Gen 2 lock command (see the EPC Gen2 specification for details). Attribute value is 4 bytes long.

$73

Devices: *all printer models with UHF module*

G2Password: a value representing the Acess / Kill password parameter for the EPC Class 1 Gen 2 commands (see the EPC Gen2 specification for details). Attribute value is 4 bytes long.

$74

Devices: *all printer models with UHF module*

G2NSI: a value representing the numbering system identifier for the EPC Class 1 Gen 2 tags' id (see the EPC Gen2 specification for details).
Attribute value is 2 bytes long.

$75

Devices: *all printer models with UHF module*

QParameter: a value representing the initial value for the Q parameter involved in the EPC Class 1 Gen 2 anticollision algorithm (see the EPC Gen2 specification for details). Attribute value is 2 bytes long.

$76

Devices: *all printer models with UHF module*

ReaderInfo: a string indicating the model and the serial number of the reader.

$77

Devices: *all printer models with UHF module*

RFRegulation: a value representing the RF regulation to use. Attribute value is 2 bytes long and can assume the following values:

0x00 = ETSI EN 302 208
0x01 = ETSI EN 300 220
0x02 = FCC
0x03 = Malaysia

Note: not all the value are supported by all the readers. For the list of RF regulation supported by each reader please refer to the reader's user manual.

$78

Devices: *all printer models with UHF module*

RFChannel: a value representing the RF channel to use. Attribute value is 2 bytes long and can assume values in the range 0 … 9. Channels are referred to the ETSI EN 302 208 regulation.

$7A

Devices: *all printer models with UHF module*

RSSI: a value representing the backscattered RF field strenght. Attribute value is 2 bytes long.

$96

Devices: *all printer models with UHF module*

PowerSet: a value representing the RF power emitted during the communication with tags. Attribute value is 4 bytes long. (used to set a new current value).

$FB

Devices: *all printer models with UHF module*

SourceName: a string representing the name of the data source. Attribute value has a maximum length of 30 bytes and can assume the following values:

“Source_0”, “Source_1”, “Source_2”, “Source_3”

4.2.2 Command codes

$12

Devices: *all printer models with UHF module*

RawReadIDs: permits to get all the tag's Ids that are under the RF field of the selected source (Deprecated).

Parameters:

	SourceNameIn:	[in] the name of the source to use.
	SourceNameOut:	[out] the name of the source used.
	ReadPointName:	[out] the name of the readpoint.
	TimeStamp:	[out] the time at which the tags are detected.
	TagIDLen:	[out] the ID length of the tags detected.
	ListOfIDs:	[out] the list of Ids detected from the source.
	ResultCode:	[out] the result code.

Note: out parameters are repeated for each readpoint in the source.

$13

Devices: *all printer models with UHF module*

NewRawReadIDs: permits to get all the tag's Ids that are under the RF field of the selected source.

Parameters:

	SourceNameIn:	[in] the name of the source to use (optional).
	Length:	[in] Filter Mask Lenght (optional).
	TagID:	[in] the Filter Mask Value (optional).
	TagAddress:	[in] Filter Mask Start Address (optional)

For each tag detected the parameters returned by the command are:

SourceNameOut:	[out] the name of the source used
ReadPointName:	[out] the name of the readpoint.
TimeStamp:	[out] the time at which the tags are detected.
TagType:	[out] the tag's type.
TagIDLen:	[out] the ID length of the tags detected.
TagID:	[out] the tag's id.
ResultCode:	[out] the result code.

Note: out parameters are repeated for each readpoint in the source.

$6E

Devices: *all printer models with UHF module*

ReadTagData: permits to read data from the tag memory.

Parameters:

SourceName:	[in] the name of the source to use.
TagIDLen:	[in] the ID length of the tag.
TagID:	[in] the ID of the tag.
TagAddress:	[in] the address from which read the data.
Length:	[in] the number of bytes to read.
TagValue:	[out] the data read from the tag memory.
ResultCode:	[out] the result code.

$6F

Devices: *all printer models with UHF module*

WriteTagData: permits to write data to the tag memory.

Parameters:		
	SourceName:	[in] the name of the source to use.
	TagIDLen:	[in] the ID length of the tag.
	TagID:	[in] the ID of the tag.
	TagAddress:	[in] the address where to write the data.
	Length:	[in] the number of bytes to write.
	TagValue:	[in] the data to write to the tag memory.
	ResultCode:	[out] the result code.

$70

Devices: *all printer models with UHF module*

LockTag: permits to lock data into the tag memory.

Parameters:		
	SourceName:	[in] the name of the source to use.
	TagIDLen:	[in] the ID length of the tag.
	TagID:	[in] the ID of the tag.
	TagAddress:	[in] the address where to write the data.
	ResultCode:	[out] the result code.

$71

Devices: *all printer models with UHF module*

RESERVED

$72

Devices: *all printer models with UHF module*

SetBitRate: permits to set the BitRate to use.

Parameters:		
	BitRate:	[in] the BitRate to set.
	ResultCode:	[out] the result code.

$74

Devices: *all printer models with UHF module*

SetProtocol: permits to set the protocol to use.
Parameters: Protocol: [in] the protocol to use.
ResultCode: [out] the result code.

$75

Devices: *all printer models with UHF module*

RESERVED

$79

Devices: *all printer models with UHF module*

GetProtocol: permits to get the protocol in use.
Parameters: Protocol: [out] the protocol in use.
ResultCode: [out] the result code.

$7B

Devices: *all printer models with UHF module*

SetDESB: permits to enable or disable the “Data Exchange Status Bit” handling during the anti-collision algorithm when ISO 18000-6b air protocol is in use.
Parameters: DESBEnable: [in] enable/disable value.
ResultCode: [out] the result code.

$7C

Devices: *all printer models with UHF module*

GetFirmwareRelease: permits to get the firmware revision.
Parameters: FWRelease: [in] the firmware release.
ResultCode: [out] the result code.

$7D

Devices: *all printer models with UHF module*

GetDESB: permits to get the current setting of the “Data Exchange Status Bit” handling.

Parameters:		
	DESBStatus:	[in] enabled/disabled value.
	ResultCode:	[out] the result code.

$80

Devices: *all printer models with UHF module*

RFOnOff: permits to start/stop the generation of a continuous wave.
Used only for test and measurements purposes.

Parameters:		
	RFOnOff:	[in] = 0 → stop; != 0 → start
	ResultCode:	[out] the result code.

$81

Devices: *all printer models with UHF module*

GetBitRate: permits to get the BitRate in use.

Parameters:		
	BitRate:	[out] the BitRate in.
	ResultCode:	[out] the result code.

$82

Devices: *all printer models with UHF module*

BlockWriteTag: permits to write data to the tag memory. This function uses the ISO18000-6b Write4Byte command to speed up the writing of large amount of data at one time.

Parameters:		
	SourceName:	[in] the name of the source to use.
	TagIDLen:	[in] the ID length of the tag.
	TagID:	[in] the ID of the tag.
	TagAddress:	[in] the address where to write the data.
	Length:	[in] the number of bytes to write.
	TagValue:	[in] the data to write to the tag memory.
	ResultCode:	[out] the result code.

$83

Devices: *all printer models with UHF module*

SetRS232: permits to modify the settings of the serial port.

Parameters:		
	Baudrate:	[in] the baud rate value.
	Databits:	[in] the data bits setting.
	Stopbits:	[in] the stop bits setting.
	Parity:	[in] the parity setting.
	Flowctrl:	[in] the flow control setting.
	ResultCode:	[out] the result code.

$84

Devices: *all printer models with UHF module*

SetDateTime: permits to modify date and time.
Parameters: Datetime: [in] the date and time to set up.
ResultCode: [out] the result code.

$85

Devices: *all printer models with UHF module*

GroupSelectUnselect: permits to execute the tag selection commands defined by the ISO18000-6B protocol.
Parameters: SourceName: [in, optional] the name of the source to use.
Operation: [in] the tag selection operation.
Bytemask: [in] the byte mask as defined by the protocol.
TagAddress: [in] the address where to compare the data.
TagValue: [in] the data to compare with the tag memory.
TagID: [out] the ID of the tag.
ResultCode: [out] the result code.

$86

Devices: *all printer models with UHF module*

GetIO: permits to read the current status of the I/O lines.
Parameters: IORegister: [out] the status of the I/O lines.
ResultCode: [out] the result code.

$87

Devices: *all printer models with UHF module*

SetIO: permits to set the level of the output lines.
Parameters: IORegister: [in] the value to set to the output lines.
ResultCode: [out] the result code.

$88

Devices: *all printer models with UHF module*

SetIODirection: permits to define the direction of the I/O lines.
(0 = input; 1 = output)
Parameters: IORegister: [in] the direction to set to the I/O lines.
ResultCode: [out] the result code.

$89

Devices: *all printer models with UHF module*

GetIODirection: permits to read the current status of the I/O lines. (0 = input; 1 = output)

Parameters:		
	IORegister:	[out] the direction of the I/O lines.
	ResultCode:	[out] the result code.

$94

Devices: *all printer models with UHF module*

E119ProgramID: permits to write the EPC into a EPC1.19 tag.

Parameters:		
	SourceName:	[in] the name of the source to use.
	TagID:	[in] the old EPC of the tag.
	TagValue:	[in] the EPC to write into the tag memory.
	ResultCode:	[out] the result code.

$95

Devices: *all printer models with UHF module*

G2ProgramID: permits to write the EPC in a Class 1 Gen 2 tag.

Parameters:		
	SourceName:	[in] the name of the source to use.
	TagIDLen:	[in] the ID length of the tag (must be an even number).
	TagID:	[in] the EPC to write into the tag memory.
	G2NSI:	[in] the EPC numbering system.
	G2Password:	[in] the EPC Access password (optional).
	ResultCode:	[out] the result code.

$96

Devices: *all printer models with UHF module*

G2Read: permits to read data from anyone of the Gen2 tag memory banks.

Parameters:		
	SourceName:	[in, optional] the name of the source to use.
	TagIDLen:	[in] the ID length of the tag.
	TagID:	[in] the ID of the tag.
	MemoryBank:	[in] the memory bank.
	TagAddress:	[in] the address where to read the data.
	Length:	[in] the number of bytes to read (must be an even number).
	TagValue:	[out] the data read from the tag memory.
	G2Password:	[in] the EPC Access password (optional).
	ResultCode:	[out] the result code.

$97

Devices: *all printer models with UHF module*

G2Write: permits to write data into anyone of the Gen2 tag memory banks.

Parameters:		
	SourceName:	[in, optional] the name of the source to use.
	TagIDLen:	[in] the ID length of the tag.
	TagID:	[in] the ID of the tag.
	MemoryBank:	[in] the memory bank.
	TagAddress:	[in] the address where to write the data.
	Length:	[in] the number of bytes to write (must be an even number).
	TagValue:	[in] the data to write to the tag memory.
	G2Password:	[in] the EPC Access password (optional).
	ResultCode:	[out] the result code.

$97

Devices: *all printer models with UHF module*

G2Lock: permits to execute the tag lock command defined by the EPC Class 1 Gen 2 protocol.

Parameters:		
	SourceName:	[in, optional] the name of the source to use.
	TagIDLen:	[in] the ID length of the tag.
	TagID:	[in] the ID of the tag.
	G2Payload:	[in] the lock payload.
	G2Password:	[in] the EPC Access password (optional).
	ResultCode:	[out] the result code.

$99

Devices: *all printer models with UHF module*

G2Kill: permits to execute the tag kill command defined by the EPC Class 1 Gen 2 protocol.

Parameters:		
	SourceName:	[in, optional] the name of the source to use.
	TagIDLen:	[in] the ID length of the tag.
	TagID:	[in] the ID of the tag.
	G2Password:	[in] the kill password.
	ResultCode:	[out] the result code.

$9A

Devices: *all printer models with UHF module*

G2Query: permits to execute the tag query command defined by the EPC Class 1 Gen 2 protocol. If a tag is in the field result code is ERROR_SUCCESS (0x00) else result code is ERROR_TAGNOT-PRESENT (0xCA).

Parameters:		
	SourceName:	[in] the name of the source to use.
	ResultCode:	[out] the result code.

$9B

Devices: *all printer models with UHF module*

G2SetQ: permits to change the initial value of the Q parameter used in the Gen2 anticollision algorithm.

Parameters:		
	QParameter:	[in] the value of the Q parameter.
	ResultCode:	[out] the result code.

$9C

Devices: *all printer models with UHF module*

G2GetQ: permits to read the initial value of the Q parameter used in the Gen2 anticollision algorithm.

Parameters:		
	QParameter:	[out] the value of the Q parameter.
	ResultCode:	[out] the result code.

$9D

Devices: *all printer models with UHF module*

G2QueryAck: permits to execute the tag query and ack command defined by the EPC Class 1 Gen 2 protocol. If a tag is in the field result code is ERROR_SUCCESS (0x00) and the command returns the EPC code stored in the tag else the result code is ERROR_TAGNOTPRESENT (0xCA).

Parameters:		
	SourceName:	[in] the name of the source to use.
	TagID:	[out] the ID of the tag.
	ResultCode:	[out] the result code.

$9E

Devices: *all printer models with UHF module*

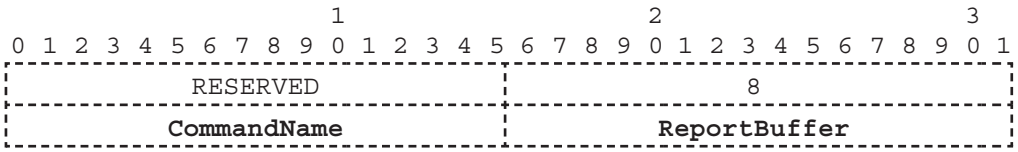
GetReaderInfo: permits to read some information about the reader itself.

Parameters:		
	ReaderInfo:	[out] a string with information about the reader.
	ResultCode:	[out] the result code.

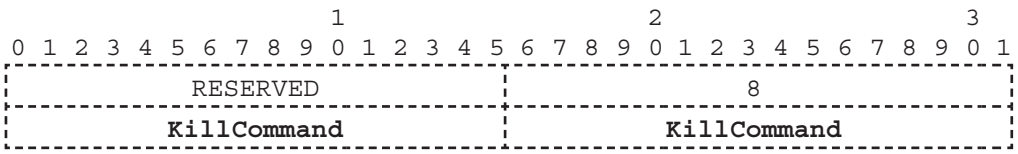
4.3 ASYNCHRONOUS NOTIFICATION: PROTOCOL SPECIFICATION

The notification channels are implemented only with sockets.
All the messages notifications are composed by a header and a body. In all cases the body of the message is a list of attribute-value pairs. The first AVP of the body is fixed and called NotifyMessage.
All the packets for notification channel share the same header format of other packet as described at § 3.
The first AVP (NotifyMessage) is followed by a list of AVPs, the number of which depends on how many tags should be notified. Each AVP has the same format of the AVP described in § 3.

The NotifyMessage has the following fixed format:



After the NotifyMessage AVP we can receive:
a) a list of AVPs (as described in table 3) followed by an AVP with 'Attribute Type' ResultCode
b) a single AVP called KillMessage with the following fixed format:



Attribute types: Notification AVP List:

Description
TimeStamp: the timestamp of the notification
TagIDLen: the ID length of the tag.
TagID: the ID of the tag.
SourceName: the name of the source to use.
EventType: the type of the notified event

4.4 DEFAULT CONFIGURATION

The default composition of sources is the following:

Source	Readpoints
Source_0	Ant0

4.5 SAMPLES

In the following sample a RawReadIDs command is examined together with the response coming from the reader.

Command sent:

```
0x8001        (Fixed)
0x0000        (Message ID)
0x00005358          (Vendor ID = CAEN SpA)
0x0021              (Message Length)
0x0000              (Reserved)
0x0008              (AVP Length)
0x0001              (AVP Type = CommandName)
0x0012              (AVP Value = RawReadIDs)
0x0000              (Reserved)
0x000F              (AVP Length)
0x00FB              (AVP Type = SourceName)
"Source_0"    (AVP Value)
```

Response received:

```
0x0001        (Fixed)
0x0000        (Message ID)
0x00005358    (Vendor ID = CAEN SpA)
0x0066        (Message Length)
0x0000        (Reserved)
0x0008        (AVP Length)
0x0001        (AVP Type = CommandName)
0x0012        (AVP Value = RawReadIDs)
0x0000        (Reserved)
0x000F        (AVP Length)
0x00FB        (AVP Type = SourceName)
"Source_0"    (AVP Value)
0x0000        (Reserved)
0x000B        (AVP Length)
0x0022        (AVP Type = ReadPointName)
"Ant0"        (AVP Value)
0x0000        (Reserved)
0x000E        (AVP Length)
0x0010        (AVP Type = TimeStamp)
0x00000578          (AVP Value = Thu Jan 1 01:23:20 1970)
0x00000000    (AVP Value)
0x0000        (Reserved)
0x0008        (AVP Length)
0x000F        (AVP Type = TagIDLen)
0x0008              (AVP Value)
0x0000        (Reserved)
0x000E        (AVP Length)
0x0011        (AVP Type = TagID)
0x05629FFF          (AVP Value)
0xC0113001          (AVP Value)

0x0000        (Reserved)
0x000E        (AVP Length)
0x0011        (AVP Type = TagID)
0xE0040F0E          (AVP Value)
0x06010000          (AVP Value)
0x0000        (Reserved)
0x0008        (AVP Length)
0x0002        (AVP Type = ResultCode)
0x0000              (AVP Value = Success)
```

5 COMMANDS INDEX

HF CUSTOM COMMANDS

$1B $3D 6
$1E $43 6
$1E $44 7
$1E $45 7
$1E $49 8
$1E $52 8
$1E $57 9
$1E $61 9
$1E $72 10
$1E $73 10
$1E $FF 11
$1F $30 11
$1F $34 12
$1F $43 12
$1F $45 13
$1F $49 13
$1F $4C 14
$1F $52 14
$1F $53 15
$1F $58 15
$1F $61 16
$1F $62 16
$1F $63 17
$1F $64 17
$1F $68 18
$1F $6B 18
$1F $71 19
$1F $72 19
$1F $73 20
$1F $75 20
$1F $76 21
$1F $77 21

HF ASK COMMANDS

$00 $01 36
$00 $04 36
$00 $06 37
$00 $07 38
$00 $08 39
$01 $01 40
$01 $03 41
$01 $04 45
$01 $05 46
$01 $0E 47
$01 $12 48
$01 $22 49
$01 $13 50
$01 $14 51
$01 $15 51
$01 $16 52
$01 $17 53
$01 $18 54
$01 $19 54
$01 $23 (not implemented yet) 55
$01 $24 55
$01 $25 56
$01 $26 56
$01 $27 57
$02 $01 63
$02 $06 63
$02 $02 64
$02 $0B 64
$02 $03 65
$02 $04 65
$02 $05 66
$02 $07 66
$02 $08 67
$02 $0A 67
$02 $0C 68
$02 $10 68
$02 $11 69
$02 $12 69
$03 $01 76
$03 $02 77
$03 $02 $04 78
$03 $03 79
$03 $04 79
$03 $05 80
$03 $07 80
$03 $06 81
$03 $08 81
$03 $09 82
$03 $0A 82
$03 $0B 83
$03 $0C 84
$03 $0E 85
$03 $0D 86
$03 $13 87

$03 $0F 88
$03 $10 89
$03 $11 90
$03 $12 90
$03 $15 91
$04 $04 92
$04 $05 92
$05 $01 94
$05 $02 94
$05 $02 $04 95
$05 $03 96
$05 $04 97
$05 $05 97
$05 $07 98
$05 $06 98
$05 $08 99
$05 $09 99
$05 $0A 100
$05 $0B 101
$05 $0C 101
$05 $10 102
$05 $11 103
$05 $12 103
$05 $14 104
$05 $15 105
$05 $12 105
$05 $0D 106
$05 $0E 107
$05 $0F 108
$05 $13 108
$05 $23 109
$05 $24 110
$06 $01 112
$06 $02 112
$06 $03 113
$06 $04 113
$06 $20 116
$06 $21 117
$06 $22 117
$06 $23 118
$06 $24 118
$06 $26 119
$06 $27 120
$06 $28 121
$06 $29 122
$06 $31 124
$06 $32 125
$06 $33 126
$06 $34 126
$06 $35 127
$06 $36 128
$10 $01 134
$10 $02 140
$10 $03 141
$10 $04 142
$10 $05 142
$10 $06 143
$10 $07 144
$10 $08 145
$10 $09 145
$10 $0D 146
$10 $0E 147
$10 $0F 148
$10 $10 148
$10 $0A 149
$10 $0B 150
$10 $0C 151
$11 $01 154
$11 $02 154
$11 $03 155
$11 $04 155
$11 $05 156
$11 $06 156
$11 $07 157
$11 $07 157
$11 $09 158
$11 $0A 158
$11 $0B 159
$11 $0C 159
$11 $0D 160
$11 $0E 160
$11 $10 161
$11 $11 161
$11 $10 162
$11 $13 162
$11 $14 163
$11 $15 163
$11 $16 164
$11 $17 164
$11 $18 165
$11 $19 165
$11 $1A 166
$11 $1B 166
$11 $1C 167
$11 $1D 167
$11 $1E 168
$11 $1F 168
$11 $20 169
$11 $21 169
$11 $22 170
$11 $23 170
$11 $24 171
$11 $25 171
$11 $26 172
$11 $26 172
$12 $01 176
$12 $02 176
$12 $03 177
$12 $04 177
$12 $05 178
$12 $06 178
$12 $07 179
$12 $08 179

UHF RFID COMMANDS

$01 190
$02 190
$0E 190
$0F 190
$10 190
$11 191
$12 191
$1E 191
$1F 191
$20 191
$21 192
$22 192
$4D 192
$4E 192
$4F 192
$50 193
$51 193
$52 193
$53 193
$54 193
$56 194
$57 194
$58 194
$59 194
$5A 194
$5B 195
$5C 195
$5D 195
$5E 195
$5F 195
$60 196
$61 196
$62 196
$63 196
$64 196
$65 197
$66 197
$67 197
$68 197
$69 197
$6A 198
$6B 198
$6C 198
$6D 198
$6E 198
$6F 199
$70 199
$71 199
$72 199
$73 199
$74 200
$75 200
$76 200
$77 200
$78 200
$7A 201
$96 201
$FB 201
$12 202
$13 202
$6E 202
$6F 203
$70 203
$71 203
$72 203
$74 204
$75 204
$79 204
$7B 204
$7C 204
$7D 205
$80 205
$81 205
$82 205
$83 205
$84 206
$85 206
$86 206
$87 206
$88 206
$89 207
$94 207
$95 207
$96 207
$97 208
$97 208
$99 208
$9A 208
$9B 209
$9C 209
$9D 209
$9E 209

Part Number : DOMC-0002E Rev. 1.00

M.U.R.S.T.
Ministry University
Research Scientific
Technology
Authorized laboratory
no.50846ZYZ

CUSTOM ENGINEERING SPA
World Headquarters
Via Berettine, 2 - 43010 Fontevivo, Parma ITALY
Tel. +39 0521 680111 - Fax +39 0521 610701
info@custom.biz - www.custom.biz

All rights reserved

www.custom.biz

Always On!